Ph.D. DISSERTATION

Clock Polarity Assignment Methodologies for Designing High-Performance and Robust Clock Trees

BY

Deokjin Joo

AUGUST 2016

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Clock Polarity Assignment Methodologies for Designing High-Performance and Robust Clock Trees

BY

Deokjin Joo

AUGUST 2016

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Clock Polarity Assignment Methodologies for Designing High-Performance and Robust Clock Trees

지도교수 Taewhan Kim

이 논문을 공학박사 학위논문으로 제출함

DECEMBER 2015

서울대학교 대학원

전기 컴퓨터 공학부

Deokjin Joo

Deokjin Joo의 공학박사 학위논문을 인준함

DECEMBER 2015

위 원 장	Kiyoung Choi
부위원장	Taewhan Kim
위 원	Ki-Seok Chung
위 원	Andrew B. Kahng
위 원	Jaeha Kim
위 원	Woohyun Paik

Abstract

In modern synchronous circuits, the system relies on one single signal, namely, the clock signal. All data sampling of flip-flops rely on the timing of the clock signal. This makes clock trees, which deliver the clock signal to every clock sink in the whole system, one of the most active components on a chip, as it must switch without halting. Naturally, this makes clock trees a primary target of optimization for low power/high performance designs.

First, bounded skew clock polarity assignment is explored. Buffers in the clock tree switch simultaneously as the clock signal switch, which causes power/ground supply voltage fluctuation. This phenomenon is referred to as clock noise and brings adverse effects on circuit robustness. Clock polarity assignment technique replaces some of the buffers in the clock trees with inverters. Since buffers draw larger current at the rising edge of the clock while inverters draw larger current at the falling edge, this technique can mitigate peak noise problem at the power/ground supply rails.

Second, useful skew clock polarity assignment method is developed. Useful clock skew methodology allows consideration of individual clock skew restraints between each clock sinks, allowing further noise reduction by exploiting more time slack. Through experiments with ISPD 2010 clock network synthesis contest benchmark circuits, the results show that the proposed clock polarity algorithm is able to reduce the peak noise caused by clock buffers by 10.9% further over that of the global skew bound constrained polarity assignment while satisfying all setup and hold time constraints.

Lastly, as multi-corner multi-mode (MCMM) design methodologies, process

variations and clock gating techniques are becoming common place in advanced technology nodes, clock polarity assignment methods that mitigate these problems are devised. Experimental results indicate that the proposed methods successfully satisfy required design constraints imposed by such variations.

In summary, this dissertation presents clock polarity assignments that considers useful clock skew, delay variations, MCMM design methodologies and clock gating techniques.

Keywords: Clock tree, Clock skew, Adjustable delay buffer, Power/ground noise, Delay variations, Multi-corner multi-mode

Student Number: 2011-30976

Contents

List of Tables

List of Figures

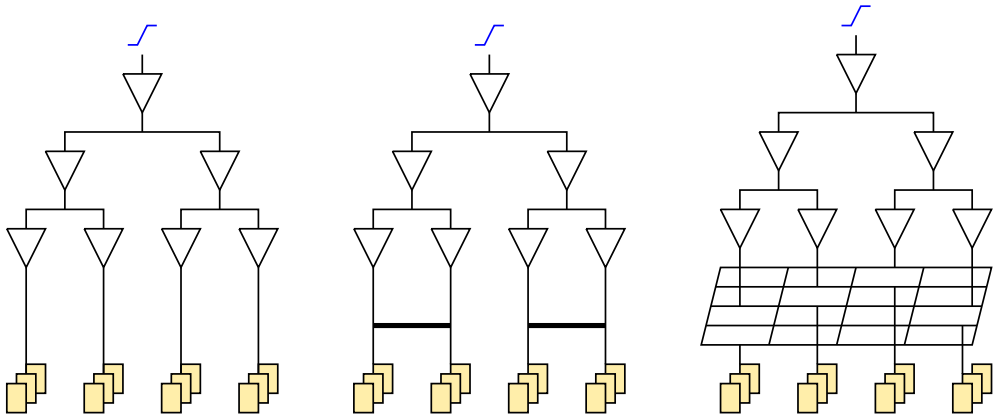
Chapter 1

Introduction

1.1 Clock Trees

In synchronous digital systems, clock distribution networks deliver the clock signal to clock sinks (i.e., flip-flops and latches). Ideally, the signal should arrive at all sinks at the same time; one of the objectives in clock network design is to control the difference among clock latencies, called the (global) clock skew, which is hard to achieve. However, in sub-45nm CMOS process technology nodes, the effect of the increased variations cause more problems to controlling the clock skews, making it a major challenge in electronic design automation (EDA).

Another major factor that shapes the clock distribution networks is the power consumption. According to the works of [1, 2, 3], clock distribution network accounts for up to 40% of the total power. Increased requirements for designing robust clock network make the power problem worse, as it may require clock signal path redundancies (in clock meshes or spines) or larger wire



(a) Clock tree (b) Clock spine (c) Clock mesh

Figure 1.1: The three types of clock distribution networks. (a) a clock tree, (b) a clock spine and (c) a clock mesh.

parameters. Fig. 1.1 is a conceptual illustration of the three clock networks: (a) is the clock tree and (b) is the clock spine, where some of the clock subtrees are linked to provide extra clock signal paths to reduce clock signal arrival time variations. Lastly, (c) is the clock mesh, where the lower level of the clock signal paths are in mesh topology. Clock meshes are most resilient to variations due to its high number of path redundancies. The higher level of this clock network is constructed with clock trees, which deliver the clock signal to parts of the clock mesh.

In this dissertation, optimization techniques and algorithms for clock trees are developed. Even though the clock trees are more susceptible to variations, it is still an essential component in constructing clock meshes and spines. Another advantage is that it consumes lower power, as it has less total capacitance than the clock spines and meshes.

1.2 Simultaneous Switching Noise

From Introduction Technology Roadmap of Semiconductors 2013 documents [4], it is predicted that many conflicting requirements such as high reliability, reduced size, high performance and lower cost will be imposed. With high operating frequencies, it is anticipated that the noise problem at the power distribution network (PDN) will continue to threaten the signal integrity of the system. One of the sources of signal integrity problem is the simultaneous switching noise (SSN) [5], which is caused by the gates clocking almost at the same time. This is especially true in synchronous circuits where all of the combinational logics connected to flip-flops (FFs) compute synchronously at the edge of the clock signal.

One of the most prominent sources of the SSN is the clock tree. When the clock tree delivers the clock signal, the clock buffers must switch between "ON" and "OFF" states simultaneously, which in turn consume power simultaneously. Since the clock tree consumes up to 40% of the total power, this implies that about 40% of the SSN in the system is emitted by the clock tree.

To mitigate the SSN problem at the clock tree, clock skew scheduling techniques have been proposed [6, 7, 8]. In [6], Benini *et al.* proposed to schedule the clock arrival times at the FFs to reduce the peak current. The clock signal arrival times are adjusted deliberately so that they switch less simultaneously. This disperses the clock noise over time. Vittal *et al.* [7] then formulated the clock arrival time scheduling problem as 0-1 integer linear program (ILP). Later, Huang, Chang and Nieh [8] refined the technique so that the computational cost of 0-1 ILP is reduced.

1.3 Clock Polarity Assignment Technique

Another technique which can mitigate the simultaneous switching noise emitted by the clock tree is the Clock Polarity Assignment technique. It has been known that selectively assigning positive or negative polarities to (initial) clock buffering elements by properly replacing some of the buffering elements with inverters and the other with buffers is an effective way of reducing the power/ground noise.[1] Fig. 1.2 illustrates the basic idea behind the polarity assignment.

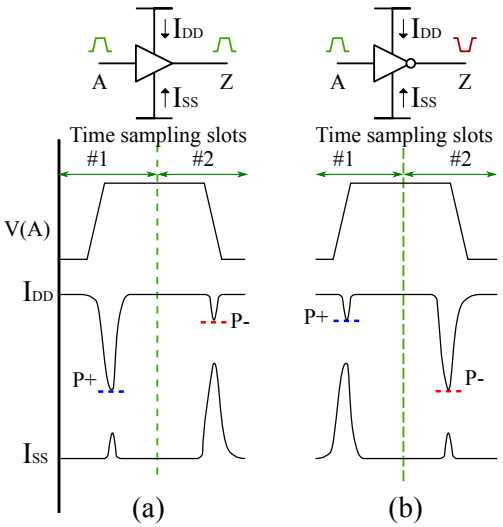


Figure 1.2: The idea behind buffer polarity assignment. (a) Buffers exhibit high $I_{\mathrm{DD}}/I_{\mathrm{SS}}$ current at rising/falling edge of clock signal, while (b) inverters emit high $I_{\mathrm{DD}}/I_{\mathrm{SS}}$ current at falling/rising edge.

A buffer is a chain of unequally sized two inverters and exhibits current noise as shown in Fig. 1.2(a); at the rising edge of clock signal, the buffer charges, drawing a high I_{DD} current while drawing a low I_{SS} current. For inverters, the opposite case happens as shown in Fig. 1.2(b). Thus, by mixing buffers and

[1] A buffering element is said to be assigned with a positive polarity or a negative polarity if its output switches in the same direction as or in the opposite direction to that of the clock source, respectively.

inverters in the buffered clock tree, the designer is able to disperse the current noise from/to $I_{\mathrm{DD}}/I_{\mathrm{SS}}$ at rising/falling edge of clock signal. This migrates some of the noise emitted in the rising/falling edge of the clock to falling/rising edge of the clock so that the noise is dispersed over time, between the clock edges. Based on this observation, several techniques of buffer polarity assignment have been proposed [9, 10, 11, 12, 13, 14, 15].

Two critical flaws of all the previous works are (1) the unawareness of the signal delay (i.e., arrival time) differences to the leaf nodes and (2) the ignorance of the effect of non-leaf nodes' current fluctuations on the total peak current waveform. Clearly, not addressing (1) and (2) in polarity assignment may cause a severely inaccurate peak current (or peak power/ground noise) estimation. By addressing the limitations, we propose a completely new solution to the problem of clock buffer polarity assignment with buffer sizing, employing a fine-grained noise estimation technique, rather than using the peak current values only at the four time sampling points of (V_{DD}, rising), (V_{DD}, falling), (GND, rising), and (GND, falling) as adopted by the previous works. In addition, we further develop an algorithm that can consider *useful clock skew* [16] for advanced systems with high performance and low power.

1.4 Contributions of this Dissertation

In this dissertation, each chapter presents clock polarity assignment algorithms and optimization techniques for solving SSN emitted by the clock trees. Specifically:

In Chapter 2, bounded clock skew polarity assignment algorithm is developed. This algorithm integrates clock polarity assignment problem with clock skew scheduling problem, which is achieved by combining clock buffer sizing technique. Moreover, this formulation allows fine-grained modelling of the clock

noise waveform, which allows better exploitation of clock skew scheduling than any other previous methods. Later in this chapter, the effects of clock polarity assignment on simultaneous switching noise is investigated with extensive experimental data. Here, the actual voltage fluctuations are observed and the properties of noise, i.e., frequency response and power spectral density, is observed.

Chapter 3 proposes an algorithm which can further improve the performance of the system, by taking useful clock skew constraints into the problem formulation. Unlike any other bounded clock skew based polarity assignment methods, the new formulation can further take advantage of individual time slack, enabling the reduction of noise compared to other known methods.

Lastly, in Chapter 4, the proposed polarity assignment algorithms are extended to cope with modern design environments, such as Multi-corner Multi-mode operating scenarios, delay variations. Further, the ease of application of polarity assignment technique is demonstrated by assigning clock polarity to gated clock trees.

Chapter 2

Clock Polarity Assignment Under Bounded Skew

2.1 Introduction

As it was introduced in Chapter 1, selectively assigning polarities to clock buffering elements is an effective way of reducing the power/ground noise. This migrates some of the noise emitted at the rising/falling edge of the clock to falling/rising edge of the clock so that the noise is dispersed over time, between the clock edges. This is illustrated by Fig. 1.2. Based on this observation, several techniques of buffer polarity assignment have been proposed [9, 10, 11, 12, 13, 14, 15].

Nieh, Huang, and Hsu [9] firstly proposed to assign positive polarity onto a half of clock buffers and negative polarity onto the remaining half of the clock buffers. Thus, they equally partitioned the clock tree into two subtrees and replaced the buffering element at the root of the subtree with an inverter, so that when the clock signal switches from 0 to 1 (or 1 to 0) all buffers on one

subtree charge (or discharge) current from V_{DD} (or to GND) while all buffers on the other subtree discharge (or charge) current to GND (or from V_{DD})[1] Even though this simple modification can reduce the total peak current over the chip upto the limit, it is not able to effectively reduce the power/ground noise in local regions. To overcome this limitation, Samanta, Venkataraman, and Hu [10] used the physical placement information of the buffering elements in determining buffers and inverters so that for local regions, roughly half of the buffering elements are assigned with positive polarity and the other half with negative polarity. Although this work is able to reduce the power/ground noise greatly, sometimes it is likely to cause a large clock skew because the effect of the different delays of inverters and buffers on the clock skew have not been taken into account. Chen, Ho, and Hwang [11] observed that the peak current occurs at the time when the clock signal arrives at the buffering elements (i.e., leaves) that are directly incident to FFs, as validated by SPICE simulation. Thus, they proposed a method of assigning polarities to the leaves, using the physical placement information of the leaves, with the objective of minimizing the power/ground noise while satisfying a minimum clock skew constraint. In addition, the approach by Ryu and Kim [12] placed more weight on the power/ground noise minimization than the clock tree embedding, thus performed polarity assignment followed by clock tree construction. However, this approach required wire overhead, which is about 5%. Kang and Kim [13] considered the delay variations in the polarity assignment. They performed polarity assignment which minimizes the power/ground noise while meeting the skew yield constraint. Jang and Kim [14] proposed an integrated approach to

[1]The buffering elements directly connected to FFs are called *leaf buffering elements* or *leaf nodes* and the other buffering elements are *non-leaf nodes*. Thus, the FFs connected to a leaf buffering elements assigned with negative polarity should be replaced with negative-edge triggered FFs.

the polarity assignment with buffer sizing to further explore the design space. Lu and Taskin [15] performed the polarity assignment to non-leaf buffering elements as well as leaves. They reduced the peak current by using polarity assignment to non-leaf nodes by 5.5% further, but the clock skew is significantly sacrificed.

The two critical flaws of all the previous works [9, 10, 11, 12, 14, 15] are (1) the unawareness of the signal delay (i.e., arrival time) differences to the leaf nodes and (2) the ignorance of the effect of non-leaf nodes' current fluctuations on the total peak current waveform. Clearly, not addressing (1) and (2) in polarity assignment may cause a severely inaccurate peak current (or peak power/ground noise) estimation. By addressing the limitations, we propose a completely new solution to the problem of clock buffer polarity assignment with buffer sizing, employing a fine-grained noise estimation technique, rather than using the peak current values only at the four time sampling points of (V_{DD}, rising), (V_{DD}, falling), (GND, rising), and (GND, falling) as adopted by the previous works. In this chapter, the new problem formulation with fine-grained noise model is proposed and an algorithm to tackle this problem is presented.

2.2 Motivational Example

The leaf buffering elements, which have no other buffering elements as their descendants, are the major contributor to the total peak current noise due to their numbers, as it had been demonstrated by [11] and illustrated in Fig. 2.1. Thus, this work focuses on assigning polarity only to the leaf buffering elements. This section illustrates how the previous works on polarity assignment lack the accuracy in estimating peak current.

Let us consider the problem of assigning polarity to the four leaf nodes on the clock tree in Fig. 2.2(a). The table in Fig. 2.2(b) shows all possible polarity

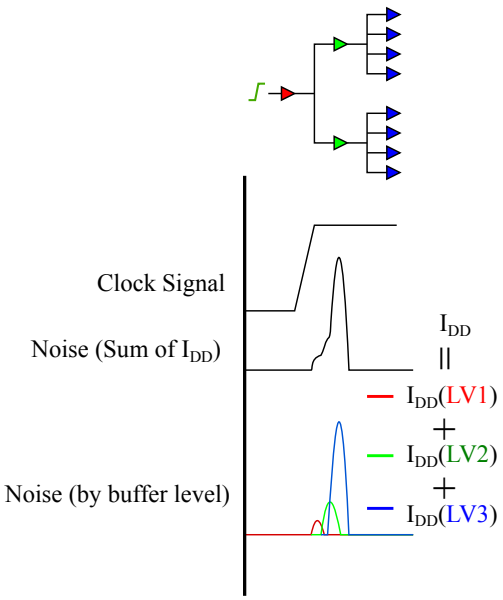
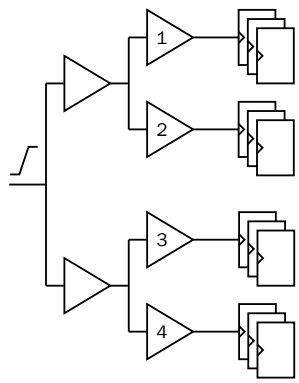


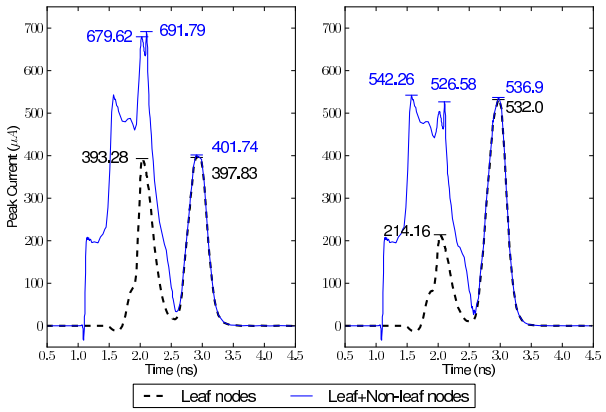
Figure 2.1: The leaf buffering elements in the clock tree out-number the non-leaf buffering elements, making them the major source of the noise in the clock tree.



(a)

Node Polarity				Peak (μA)
1	2	3	4	
P	P	P	P	781
N	P	P	P	584
P	N	P	P	584
N	N	P	P	**387**
			...	
N	N	P	N	529
P	P	N	N	394
N	P	N	N	502
P	N	N	N	499
N	N	N	N	685

(b)



(c) (d)

Figure 2.2: (a) A simple clock tree with four leaf nodes. (b) Its expected peak current value. The fourth assignment (N, N, P, P) produces the lowest value of total peak current of $387\mu A$. (c) Current waveforms by non-leaf nodes' noise unaware optimal polarity assignment ($= (N, N, P, P)$ in (b)) to leaf nodes. Dark dotted line is the current waveform from leaf nodes only while blue solid line shows the total current from all clock nodes. (d) Current waveforms resulting from non-leaf nodes' noise aware optimal polarity assignment ($= (N, N, P, N)$ in (b)) to leaf nodes.

assignments obtainable by replacing each node with buffer or inverter. Their corresponding values of total peak current can be computed by summing the peak current values of each node, according to their polarities, where P and N indicate positive and negative polarities, respectively.

From the table, we can see that the fourth assignment (N, N, P, P) produces the lowest value of total peak current, which is $387\mu A$. The dark dotted curve in Fig. 2.2(c) shows the accumulated current waveform of the leaf nodes for the polarity assignment (N, N, P, P). On the other hand, the blue solid curve in Fig. 2.2(d) shows the accumulated current waveform of *all* nodes including the two non-leaf buffers, from which we can see that the actual value of total peak current is unbalanced, i.e., skewed to the left (at time = 2.2 ps), resulting in the peak current of $691.79\mu A$. However, the dark dotted curve in Fig. 2.2(d) shows the current waveform of the leaf nodes when the polarity assignment is (N, N, P, N), thus the peak is skewed to the right. The blue solid curve in Fig. 2.2(d) which shows the resulting waveform of *all* nodes however has much reduced peak current, which is around $542\mu A$. This observation implies that the current fluctuation by non-leaf nodes should be taken into account during the process of polarity assignment of leaf nodes.

Another observation from the current waveforms in Fig. 2.2(d) indicates that by knowing that some leaf nodes may switch at different times due to unequal clock signal delays, the current fluctuation by the non-leaf nodes contributes differently to the (accumulated) current waveforms at the time when the leaf nodes switch. Thus, any time instance in a certain time interval (e.g., time in $[1.0, 4.0]$ in Fig. 2.2(d)) can be a time sampling candidate at which peak current may occur.

2.3 Problem Formulation

Problem 1 (WaveMin). *(Polarity assignment/buffer sizing for peak current minimization) Given an available buffer type set B, an inverter type set I, a sub-area that holds set L of leaf buffering elements, time sampling slots S, clock skew constraint κ, find a mapping function $\phi : L \mapsto \{B \cup I\}$ that minimizes the quantity of*

$$\max_{s \in S} \left\{ \sum_{e_i \in L} noise(\phi(e_i), s) \right\} \tag{2.1}$$

$$s.t\ t_{skew}(\phi) \leq \kappa$$

where $t_{skew}(\phi)$ is the clock skew induced by mapping ϕ and $noise(\phi(e_i), s)$ is the value of peak current estimation at time sampling point s caused by the switch of node e_i when it is assigned with type $\phi(e_i)$. $noise(\phi(e_i), s)$ is assumed to be independent of the mapping choice of $\phi(e_j)$, $i \neq j$.

Note that the set of time sampling slots S not only represents the discrete sampling times of interest, such as the rising and falling edges of the clock signal, but also the power line of interest, V_{DD} and Gnd. For example, S may have four slots, at V_{DD} and Gnd, each of which having time samples at the rising and the falling edge of the clock. As the size of S increases by including more (meaningful) time sampling points, the peak current estimation would be more accurate.

In the following, we show that the decision version of WaveMin problem, decision-WaveMin, is NP-complete, by showing that the Partition problem, which is a well-known NP-complete problem, reduces to decision-WaveMin.

Problem 2 (decision-WaveMin). *For a* WaveMin *instance with (L, B, I, S, κ) and a constant c, is there a mapping ϕ such that the value of (2.1) is less than or equal to c?*

Problem 3 (Partition). *For a finite set A and a 'size' $s(a) \in \mathbf{Z}^+$ for each $a \in A$, where $\mathbf{Z}^+$ is the set of positive natural numbers, is there a subset $A' \subseteq A$ such that*

$$\sum_{a \in A'} s(a) = \sum_{a \in A - A'} s(a)?$$

Theorem 1. DECISION-WAVEMIN *is NP-complete.*

Proof. Firstly, DECISION-WAVEMIN in NP. When a problem instance and a solution candidate ϕ is given, the $noise(\phi)$ computation and clock skew $t_{skew}(\phi)$ is achievable in polynomial time. Let us now map any instance $A, s(a)$ of the PARTITION into DECISION-WAVEMIN as follows, in polynomial time and space.

Let there be only one type of buffer/inverter in the library:

$$B = \{b\} \text{ and } I = \{v\}$$

Two slots are allocated in the set of time sampling slots S, i.e, $S = \{t_1, t_2\}$ and $|S| = 2$. For set of leaf buffering elements L, allocate one element e_i for each element a_i in A, so that $|L| = |A|$. Now, we define $noise$ values so that they correspond to $s(a)$ values in PARTITION:

For all $i = 1, 2, \cdots, |L|$,

$$\begin{aligned} noise(\phi(e_i) = b, t_1) &:= s(a_i) \\ noise(\phi(e_i) = b, t_2) &:= 0 \\ noise(\phi(e_i) = v, t_1) &:= 0 \\ noise(\phi(e_i) = v, t_2) &:= s(a_i) \end{aligned}$$

$\kappa := \infty$, so that e_i may be mapped freely without the clock skew constraints.

Since $|S| = 2$, (2.1) can be re-written as:

$$\begin{aligned} Noise(t_1) &= \sum_{e_i \in L} noise(\phi(e_i), t_1) \\ Noise(t_2) &= \sum_{e_i \in L} noise(\phi(e_i), t_2) \\ &\max \{Noise(t_1), Noise(t_2)\} \end{aligned} \tag{2.2}$$

Note that $Noise(t_1)$ and $Noise(t_2)$ correspond to $\sum_{a \in A'} s(a)$ and $\sum_{a \in A - A'} s(a)$ in PARTITION, respectively, due to the way that $noise$ is defined.

Finally, we define the noise constraint $c := \frac{1}{2} \sum_A s(a)$. Since $noise(\phi(e_i), t) > 0 \forall i$, for (2.2) to be less than or equal to c, $c = Noise(t_1) = Noise(t_2)$ must hold: while migrating noise from slot t_1 to t_2 decreases $Noise(t_1)$, $Noise(t_2)$ must increase; $Noise(t_1) < c$ implies $Noise(t_2) > c$ and vice versa. Hence, unless both the $Noise$ values are equal to c, (2.2) $> c$.

The solution instance found by this mapping can be converted back to the solution instance A' in $O(|L|)$ time, by adding a_i to set A' when $\phi(e_i) = b$. □

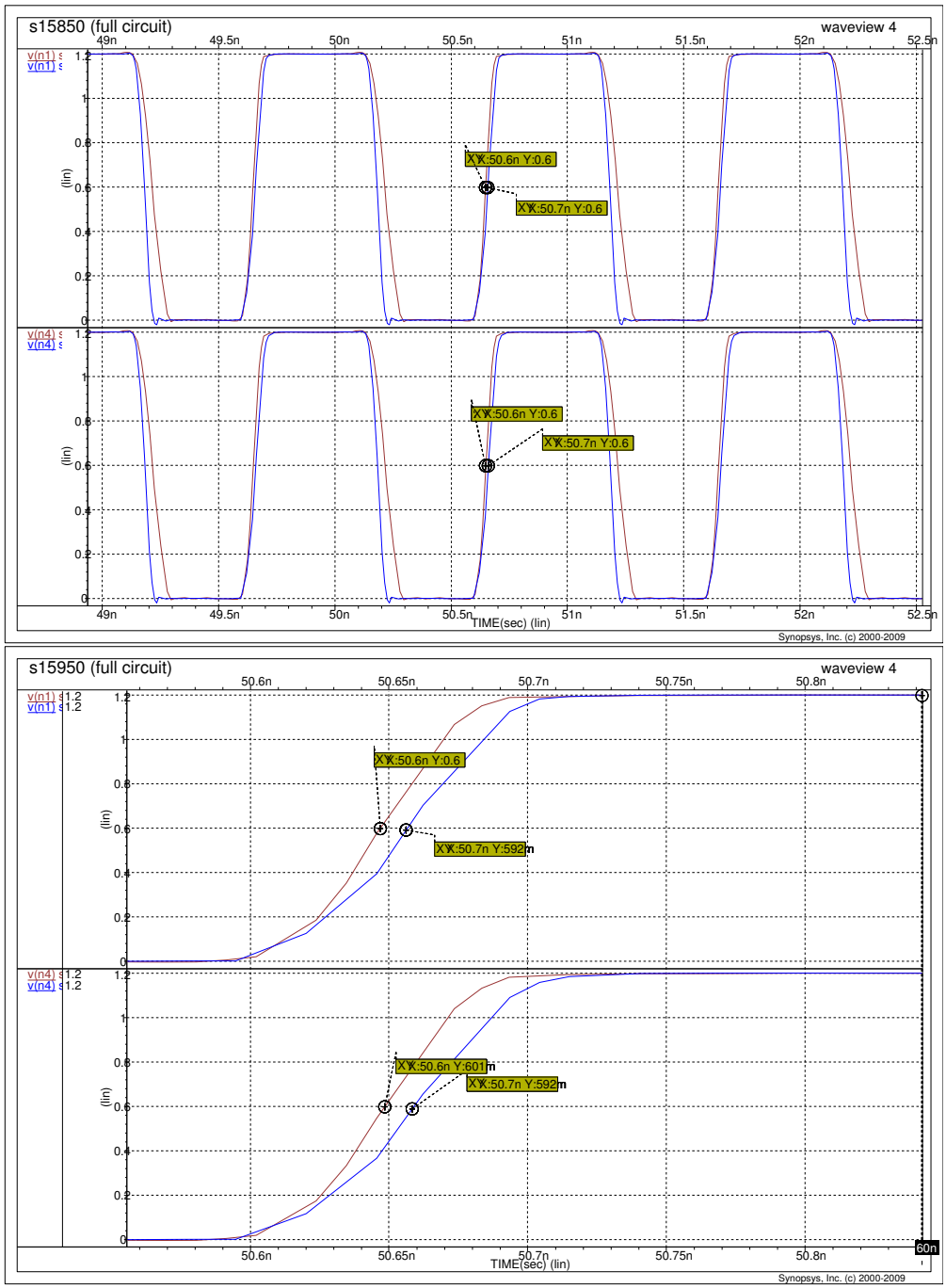


Figure 2.3: The effect of cell type change on sibling nodes. HSPICE simulations modelling ISCAS'89 benchmark circuit s15850 was executed. The circuit had 4 leaf nodes, all of which had initial cell type of BUF_X32. After replacing two of them with INV_X16, HSPICE simulation was executed again. The clock signals of the two buffers (BUF_X32) are plotted.

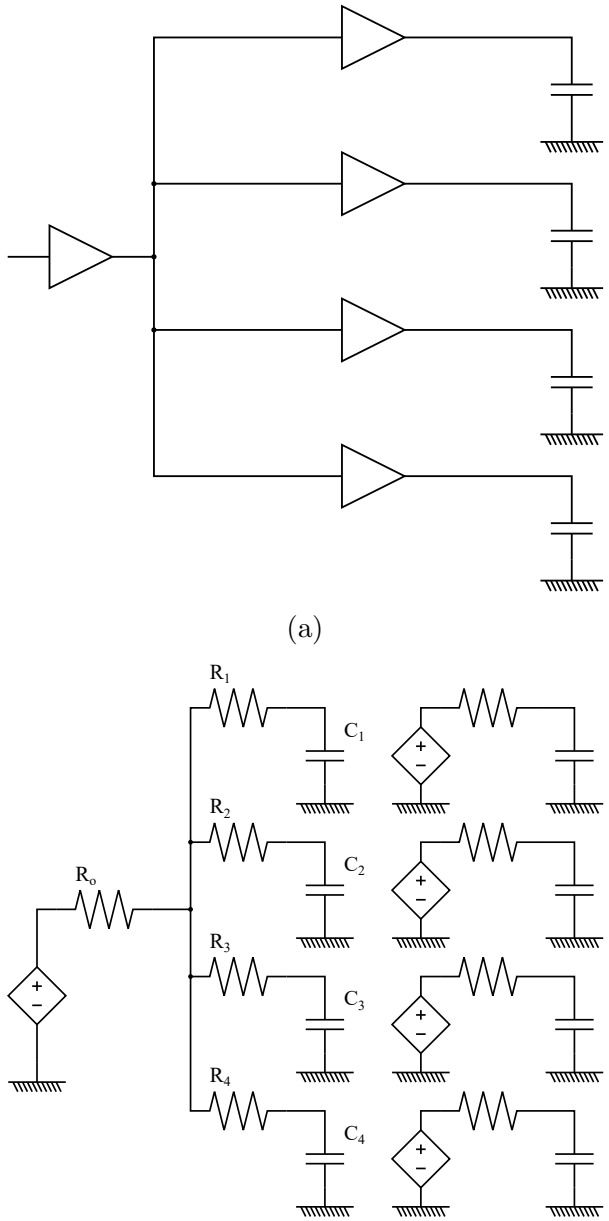


Figure 2.4: Elmore delay analysis of sibling PA/sizing. (a) The clock tree branch with 4 leaf buffering elements. (b) Simple circuit model of the tree branch.

2.4 Proposed Algorithm

2.4.1 Independence Assumption

To simplify the approach, we assume that changing the cell type of a leaf node has little impact on its sibling nodes. To verify this, HSPICE simulations modelling ISCAS'89 benchmark circuit s15850 was executed. The circuit had 4 leaf nodes, all having initial cell type of BUF_X32. After replacing two of them with INV_X16, HSPICE simulation was executed again. This is one of the worst case where half of the siblings have opposite polarity from the other half. In addition, the input capacitance of INV_X16 is 4 times that of BUF_X32 (BUF_X32 has INV_X1 at its input, in 45nm Nangate Library [19]).

For a simple analysis, the clock tree is depicted in Fig. 2.4. In (a), the clock tree is shown. Each of the 4 leaf buffering element is driving their respective loads. In (b), a simple circuit model of (a) is illustrated. R_o is the output resistance of the parent buffer. R_1, R_2, R_3 and R_4 are the resistances of the interconnection wire. C_1, C_2, C_3 and C_4 are the input capacitances of the leaf buffering elements. Since the subtrees of each leaf are isolated from the others, the error of delay computation is induced from the input side of the leaf buffers. Let us assume that the buffering elements at C_1 and C_2 are replaced by inverters and their capacitances became C_1' and C_2'. Assuming that the delay at C_1 is independent of the change in C_2, the Elmore delay at C_1 is computed as

$$R_o(C_1' + C_2 + C_3 + C_4) + R_1 C_1' \tag{2.3}$$

However, considering the change at C_2, the actual Elmore delay is

$$R_o(C_1' + C_2' + C_3 + C_4) + R_1 C_1' \tag{2.4}$$

The error is $R_o(C_2' - C_2)$. Generally, the error is related to the change in

the total capacitance at the branch, $(\Delta C_1 + \Delta C_2 + \Delta C_3 + \Delta C_4)$. This is true in more elaborate circuit models. In the HSPICE simulation, let $C_3 = C_4 = C$ for BUF_X32 and $C_1' = C_2' = 4C$ for INV_X16. Then the change in the total capacitance is $6C$. The HSPICE simulation results are plotted in Fig. 2.3. The clock signals of the two buffers (BUF_X32) are plotted. The arrival time change at the buffers are roughly 0.01 ns when the clock period is 1 ns (1 GHz). This is only 1% of the clock period. Hence, in this work, we assume that it is safe to independently optimize and change the polarity/size of each leaf buffering element. However, to prevent high difference in the total capacitance at the leaf buffer inputs, the designer must carefully choose the available buffer and inverter library sets B and I.

2.4.2 Characterization of Noise

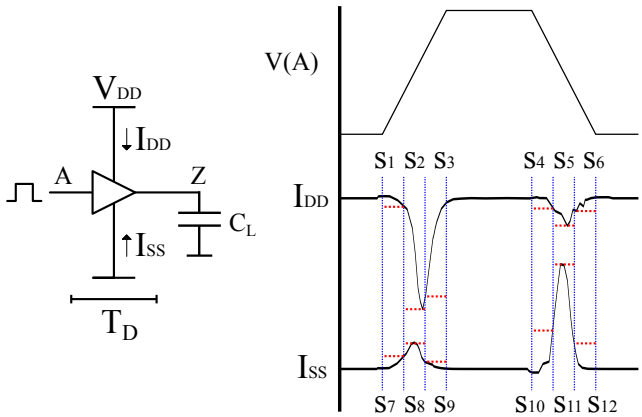


Figure 2.5: Characterizing a buffer type in B. (a) A clock pulse is applied to the input of the buffer. Then, the current waveforms of I_{DD} and I_{SS}, and the signal propagation time T_D of the buffer are measured and recorded. (b) Only the 'hot spots' of waveforms of I_{DD} and I_{SS} are captured as most of the non-zero sampled values are located near the rising and falling edges of the input. In this example, there are 12 sampling points, $s_1, s_2, \cdots, s_{12}$ and $s_1, \cdots, s_6$ are from I_{DD} and $s_7, \cdots, s_{12}$ from I_{SS}. Inverter types in I are also similarly characterized.

In the problem formulation of the previous section, an accurate characterization of the *noise* property is necessary for the solution of the problem, for it is directly required in the computation of the objective function, (2.1). For each buffer/inverter types in $|B|$ and $|I|$, the *noise* parameters is characterized as follows.

Fig. 2.5(a) shows a buffer type which is being characterized. By applying a clock pulse to the input A, the current waveforms of I_{DD} and I_{SS} and the signal propagation time T_D of the buffer are measured and recorded as a data entry in the lookup table for *noise*. C_L is varied also, as the *noise* value depends on C_L as well. During optimization, linear interpolation method is used to construct the required *noise* function, which correctly reflects the environment each leaf buffer is situated. Note that it is possible to only capture the 'hot spots' of waveforms of I_{DD} and I_{SS} since the sampled values in the current waveforms are mostly zero and the non-zero values are located near the rising and falling edges of the input. For example, in Fig. 2.5(b), times s_1, $s_2, \cdots, s_{12}$ are selected as the time sampling points to form S in (2.1).

2.4.3 Overview of the Proposed Algorithm

Fig. 2.6 shows the flow of our proposed clock polarity assignment. The inputs to our polarity assignment framework are a synthesized buffered clock tree, libraries B and I, and clock skew constraint κ, from which the preprocessing of extracting noise data and sampling points is performed.

With the input clock tree, all *feasible intervals* are extracted. Given a time t and the clock skew constraint κ, the interval $[t-\kappa, t]$ is said to be feasible if for all leaf buffering elements in L, there exists at least one mapping ϕ such at all the clock signal arrival times fall in the interval $[t-\kappa, t]$. The feasibility of an interval can be readily checked in polynomial time by checking if there exists

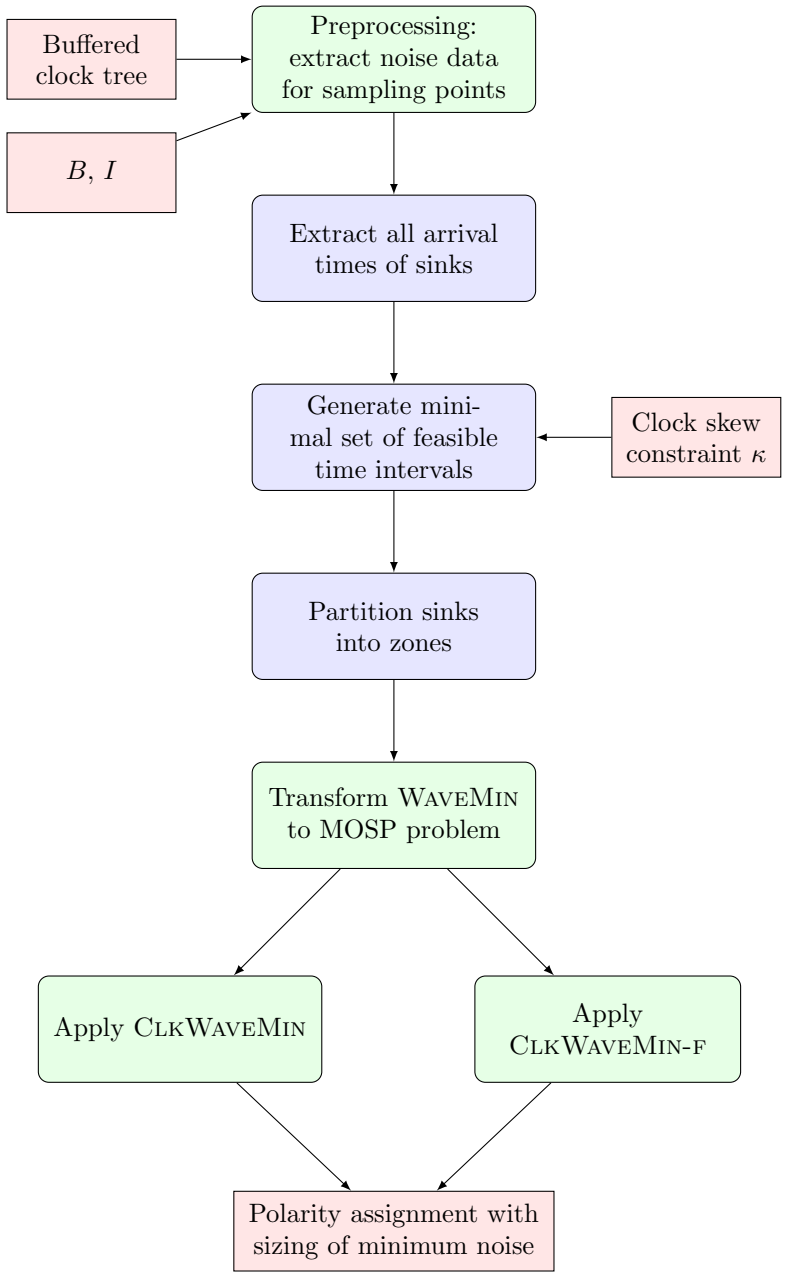


Figure 2.6: The flow of the algorithm to solve WaveMin problem.

at least one buffer/inverter type $x \in B \cup I$, such that for each leaf node $e_i \in L$, so that the arrival time $t - \kappa \leq arr(e_i, x) \leq t$. The number of feasible intervals are finite and bounded by $|L| \cdot (|B| + |I|)$, when the mapping of buffer/inverter type at each clock buffering elements generate unique arrival times for all clock sinks. This step is run globally and ensures that the clock skew bound κ is met globally.

In the next step, the clock buffering elements are partitioned into zones by their locations. Since modern designs have a large number of leaf buffering elements, it may not be feasible to solve the noise computation and minimization globally. Also, as clock noise is a local phenomenon related to the power delivery networks (PDNs), it makes sense to partition the problem with locality information. The partitioning can be done by bisection, until there are only at most N leaf buffering elements in each zone, where N is a parameter defined by the designer. Or, the designer may divide the zones by reflecting the design, e.g., by submodules. This issue is further discussed later in Section 2.4.6.

Now, the problem is solved by solving all of the subproblems defined by a zone z_i in the circuit and a time interval $[t - \kappa, t]$. For each subproblem, we minimize the noise quantity defined as (2.1). However, finding the best mapping ϕ in the zone is still a difficult task. We propose to map this problem as Multi-Objective Shortest Path (MOSP) problem. Then the problem can be solved with a fully polynomial ϵ-approximation algorithm devised by Warburton [17]. The algorithm is fully polynomial in both time and space criteria: $O(rn^3(n/\epsilon)^{2r})$ time and $O(rn(n/\epsilon)^r)$ space where r is the arc weight dimension and n is the number of vertices in MOSP graph. The formulation of the subproblem to the MOSP problem is described in the next section.

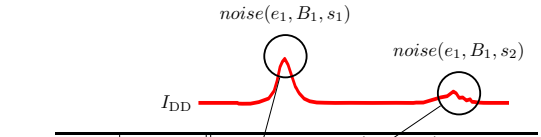
2.4.4 Mapping WaveMin Problem to MOSP problem

First, MOSP problem is formally defined.

Problem 4 (MOSP)**.** Given a directed graph $G = (V, A)$, r dimensional vector weight $w \in W(a)$ for each arc $a \in A$ and two vertices $s, t \in V$, find all Pareto-optimal paths[2] from s to t, where the cost of a path is defined as the sum of arc weights along the path.

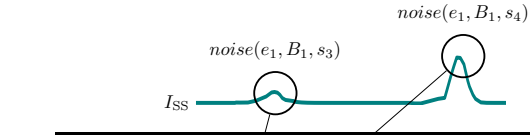
Even for $r = 2$, it is known that the decision version of MOSP problem is NP-complete [18]. Fig. 2.7 shows an example of converting an instance of WaveMin in an interval $[t_1 - \kappa, t_1]$ to a graph of MOSP problem. Column *Feasible types* in the tables in Figs. 2.7(a) and (b) are the buffers and inverters in $B \cup I$ that can be assigned to the corresponding sink in L without violating clock skew constraint, and the numbers in the entries of the tables represent the corresponding noise values of I_{DD} and I_{SS}. For example, the number (= 96) in the entry at location (e_1, B_1, s_1) in Fig. 2.7(a) indicates that the peak noise of I_{DD} at time s_1 is 96 when sink e_1 is assigned with buffer B_1, and the number (= 75) in the entry at location (e_4, I_1, s_3) in Fig. 2.7(b) indicates that the peak noise of I_{SS} at time s_3 is 75 when sink e_4 is assigned with inverter I_1. Note that the WaveMin instance has four time sampling slots $s_1, \cdots, s_4$ where s_1 and s_2 are the sampling slots for I_{DD} noise waveform and s_3 and s_4 are for I_{SS}. The transformed MOSP graph of the WaveMin instance in Figs. 2.7(a) and (b) is shown in Fig. 2.7(c). The MOSP graph has vertices with "row" (representing sinks) and "column" (representing elements in $B \cup I$) properties, and each vertex corresponds to a distinct feasible assignment of a sink to a buffer or inverter in $B \cup I$ in the WaveMin instance. For example, the vertex labelled with e_2B_2 i.e., located at the intersection of row e_2 and column B_2

[2]It corresponds to finding all non-dominated paths in the graph. That is, paths for which it is not possible to find a better total weight on a vector entry without getting worse on some of the other entries.



(a)

Node	Feasible Types	*noise* (V_{DD} rail)							
		B_1		B_2		I_1		I_2	
		s_1	s_2	s_1	s_2	s_1	s_2	s_1	s_2
e_1	B_1, B_2	96	6	82	5				
e_2	B_2, I_1			84	5	8	73		
e_3	I_2							4	72
e_4	I_1, I_2					4	79	5	77



(b)

Node	Feasible Types	*noise* (V_{DD} gnd)							
		B_1		B_2		I_1		I_2	
		s_3	s_4	s_3	s_4	s_3	s_4	s_3	s_4
e_1	B_1, B_2	3	83	5	80				
e_2	B_2, I_1			6	78	70	7		
e_3	I_2							71	4
e_4	I_1, I_2					75	7	76	6

(c)

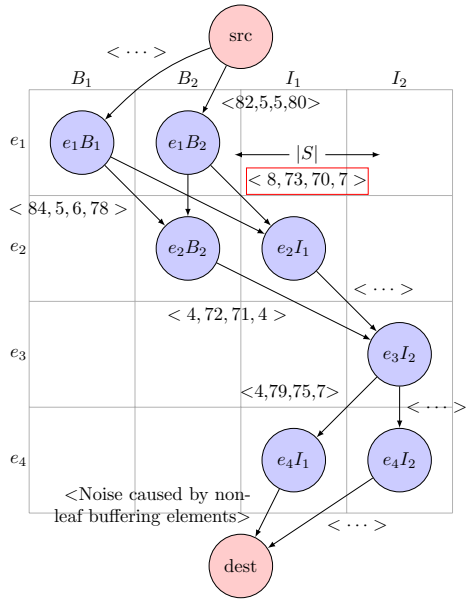


Figure 2.7: An example of converting an instance of WaveMin with interval $[t_1 - \kappa, t_1]$ to an MOSP graph. (a) Computation of *noise* for I_{DD} sampling slots, (b) computation of *noise* for I_{SS} sampling slots and (c) the converted MOSP instance of the WaveMin subproblem.

corresponds to the option of assigning sink e_2 with buffer B_2 in Fig. 2.7(a). A vertex in row i has an incoming arc from every vertex in row $i-1$. The MOSP graph has two dummy vertices called *src* and *dest*. The *src* is directed to every vertices in the first row and every vertex in the last row is directed to *dest*. For an arc (u, v) where v is at row r and column c, the arc weight is defined as $w(u, v) = (noise(e_r, c, s_1), \cdots, noise(e_r, c, s_{|S|}))$. For example, any arc directed to vertex e_2I_1 in Fig. 2.7(c) has arc weight of $w(\cdot, e_2I_1) = (noise(e_2, I_1, s_1)$, $noise(e_2, I_1, s_2)$, $noise(e_2, I_1, s_3)$, $noise(e_2, I_1, s_4)) = (8, 73, 70, 7)$, as shown in the red box in Fig. 2.7(c). One exception is vertex *dest*. For the arcs directed to *dest*, the arc weights are assigned to reflect the noise caused by the non-leaf buffering elements of the clock tree to account for observations in section 2.2. **Algorithm** 1 describes the conversion of a WaveMin instance to an MOSP graph.

The multi-dimensional distance $w(u, v)$ is assigned as the estimated noise value when $\phi(row(v)) = col(v)$, hence the distance of path $s \rightsquigarrow t$ represents the (accumulated) noise, and the vertices in between the path indicate the corresponding assignments. For example, if vertex e_2B_2 is on path $s \rightsquigarrow t$, node e_2 should be assigned with a buffer of type B_2. The degree of MOSP graph G is $O(|B|+|I|)$ since a node can have at most $|B|+|I|$ incoming and outgoing arcs. Therefore, the number of arcs in G is bounded by $O(2(|B|+|I|)|L|+2) = O(|L|)$, since there are only limited available types of buffers and inverters, meaning that $|B| + |I|$ is a constant. Lastly, arc weight dimension r equals $|S|$.

The resulting problem is solved with Warburton's algorithm [17] and all approximated Pareto-optimal paths from s to t are found. Among the retrieved paths, we take the path with the minimum worst distance as our WaveMin solution. The path is a valid solution to WaveMin problem because the MOSP graph is directed acyclic since arc (u, v) exists between vertices u

Algorithm 1 Conversion of WaveMin instance to MOSP graph.

function WaveMin 2MOSP($L, \kappa, noise, B, I, S$)
  $V \leftarrow \emptyset$; ▷ Vertices
  $A \leftarrow \emptyset$; ▷ Arcs
  for $e_i \in L$ **do** ▷ **Vertex construction**
    for *type* $\in$ feasible subset of $B \cup I$ for e_i **do**
      // Allocate and place vertices at proper place
      $v \leftarrow$ new_vertex();
      row(v) $\leftarrow i$;
      column(v) $\leftarrow$ *type*;
      $V \leftarrow V \cup \{v\}$;
    end for
  end for
  Create and prepend a row, as the new first (0-th) row;
  Place a dummy node *src* in the first row;
  for $r \in$ rows **do** ▷ **Arc construction**
    $q \leftarrow$ next_row(r);
    for all (u, v), where $u \in r$ and $v \in q$ **do**
      $a = (u, v)$;
      $A \leftarrow A \cup \{a\}$;
      type = column(v);
      // S is the set of sampling points
      weight(a) $\leftarrow$ $noise(e_r,$ *type*, $S)$;
    end for
  end for
  $r \leftarrow$ the current last row;
  Create and append a row, as the new last ((r+1)-th) row;
  Place a dummy node *dest* in the last row;
  for all vertices u in row r **do** ▷ Arcs to *dest* vertex
    Allocate and add a new arc $(u, dest)$ in A;
    weight(a) $\leftarrow$ $noise$(non-leaf, S)
  end for
  return $G(V, A)$;
end function

and v only if $row(v)$ and $row(u)$ are adjacent. The overall runtime of Warburton's approximation algorithm is given as $O(rn^3(n/\epsilon)^{2r})$ and substituting r and n yields $O(|S||L|^3((|B|+|I|)\cdot|L|/\epsilon)^{2|S|})$. The final selection of min-max solution among $O(r(n/\epsilon)^r)$ Pareto-optimal solutions has execution time of $O(r\times r(n/\epsilon)^r + r(n/\epsilon)^r) = O(|S|^2((|B|+|I|)\cdot|L|/\epsilon)^{|S|})$.

2.4.5 A Fast Algorithm

Algorithm 2 WaveMin-f: a fast greedy algorithm of WaveMin.

```
1: procedure GreedyMOSP(G(V, A))
2:     sum(S) ← noise(non-leaf,S);
3:     while |V| ≠ 0 do
4:         best(S) ← ∞;
5:         best_v ← nil;
6:         for v ∈ V do                          ▷ Get least worsening v
7:             next_sum(S) ← sum(S) + noise(v, S);
8:             if max(next_sum(S)) < max(best(S)) then
9:                 best(S) ← next_sum(S);
10:                best_v = v;
11:            end if
12:        end for
13:        e_i ← row(best_v);
14:        y ← col(best_v);                      ▷ y: feasible buffer or inverter
15:        Remove nodes in row e_i from V;
16:        Assign leaf node e_i with y;
17:        sum(S) ← best(S);
18:    end while
19: end procedure
```

In addition to using Warburton's approximation algorithm, we propose a fast version WaveMin-f with lower time and space complexity, as presented in **Algorithm** 2. In contrast to WaveMin which tries to find an optimal or approximate shortest path, WaveMin-f performs the polarity assignment vertex by vertex basis iteratively, by selecting and assigning a buffer or an inverter

with the "least noise-worsening" first from its current state. Let *sum* denote the *noise* expectation contributed by the currently selected set of vertices in the MOSP graph $G(V, A)$ as well as all the non-leaf nodes in the clock tree. Then, for each unselected vertex $v \in V$, $M(v) = max(sum(s_i) + noise(v, s_i), s_i \in S)$ is calculated and the vertex with the minimum $M(v)$ is selected as the vertex of choice in this iteration. For next iteration, *sum* is updated and the other vertices in the same row as v are removed from V to prevent the leaf node associated to v from further sizing or polarity assignment. The iteration continues until there is no more vertex in V. The space used by WaveMin-f is $O(|S||L|)$ since there are $O(|L|)$ vertices in the MOSP graph and the running time is $O(|S||L|^2)$.

2.4.6 Zone Sizing/Partitioning Method

Zone partitioning was introduced into WaveMin as a heuristics to solve the problems on large circuits by dividing the problem into smaller subproblems. The method of partitioning can affect the optimization results. It is reported in [14] that larger zones lead to better optimization results, since the optimizer can take more leaf buffering elements into the scope of optimization.

Empirically, when each zone had roughly 5 to 10 nodes, this led to good enough optimization results. However, as zone size increases, the gain saturates and the size of subproblem instance becomes large, which in turn increases the optimization time. Another factor to consider is that noise is a local phenomenon affecting the PDN since one buffer can influence only a limited region of the PDN. Having excessively large zones will decorrelate the peak current metric from the actual V_{DD}/Gnd voltage fluctuation.

Note that it is possible to optimize only the critical subregions of the chip specified by the designer. Also, when the designer knows the peak current bud-

get, the designer can provide this as an input to WaveMin to terminate the optimization when budget is met.

2.5 Experimental Results

2.5.1 Experimental Setup

The proposed algorithms WaveMin and WaveMin-f have been implemented in C++ language on a Linux machine and tested on ISCAS'89 benchmark circuits. The benchmarks were synthesized using Synopsys' Design Compiler and clock trees were synthesized as zero skew trees (<10 ps clock skew in HSPICE simulations) with Synopsys' IC Compiler, using Nangate 45nm Open Cell Library [19]. RC extractions were performed on IC Compiler and HSPICE simulation was done on the clock trees. In addition, to synthesize ISPD 2009 Clock Tree Synthesis contest benchmarks, we have employed the algorithm in [20].

We also implemented the best known bounded clock skew polarity assignment algorithm PeakMin [14] for the comparison with our algorithms. All leaf nodes were attempted to be assigned to any of BUF_X8, BUF_X16, INV_X8, and INV_X16. The benchmark circuits were partitioned into a square grid of zones, where the grid size had been determined empirically as $10 \times 10 \mu m^2$. On average, each zone contained 4.3 nodes for ISCAS'89 benchmarks and 4.9 nodes for ISPD'09 benchmarks. In particular, benchmark design s35932 has 7.1 nodes in each zone on average.

2.5.2 Noise Reduction

Table 2.1 summarizes the comparison of the results produced by PeakMin [14] and WaveMin when clock skew bound is set to $\kappa = 20$ ps. V_{DD} and Gnd noises are the maximum voltage fluctuations observed in the power and ground

Table 2.1: Comparison of results by PeakMin [14] and WaveMin when $\kappa = 20$ ps, $\epsilon = 0.01$, $|S| = 158$. The column n denote the total number of buffering elements, including both non-leaf nodes and leaf nodes and $|L|$ is the number of leaf buffering elements.

Bench-mark Circuit	n	$\|L\|$	Peak current (mA)		
			PeakMin [14]	WaveMin	Improvement (%)
s13207	58	50	6.45	7.25	-12.39
s15850	22	19	3.01	3.01	0.00
s35932	323	246	21.59	15.59	27.79
s38417	304	228	19.83	11.88	40.09
s38584	210	169	16.92	11.58	31.56
ispd09f31	328	111	75.50	62.17	17.66
ispd09f34	210	69	49.12	46.85	4.62
Average					**15.62**

grids, respectively. In summary, WaveMin reduces the peak current by 15.6% on average.

Table 2.2 shows comparison with results by WaveMin using various time sampling points and our fast WaveMin-f ($|S| = 158$). For $|S| = 4$, from I_{SS} and I_{DD} waveform, two values from each current profile were obtained by extracting the maximum value from the first and the second halves of the waveform. We can see that the use of more sampling points leads to a further reduction in peak current. Further, our fast greedy algorithm WaveMin-f produces result close to that by WaveMin with 158 sampling points, but run time is significantly fast.

2.5.3 Simulation on Full Circuit

To isolate the effects of the algorithm on the reduction of the peak current emitted by the clock trees, HSPICE simulations in the previous section was run on clock tree circuits without other combinational logics that compose the

Table 2.2: Comparison with WaveMin ($\epsilon = 0.01$) varying the number of time points and WaveMin-f ($|S| = 158, \kappa = 20$ ps).

Benchmark circuit	PeakMin		WaveMin						WaveMin-f	
			$\|S\| = 4$		$\|S\| = 8$		$\|S\| = 158$		$\|S\| = 158$	
	Peak curr. (mA)	Exec. time (ms)	Peak curr. (mA)	Exec. time (ms)	Peak curr. (mA)	Exec. time (ms)	Peak curr. (mA)	Exec. time (ms)	Peak curr. (mA)	Exec. time (ms)
s13207	6.5	<0.01	7.2	<0.01	7.2	<0.01	7.2	<0.01	7.25	<0.01
s15850	3.0	<0.01	3.0	<0.01	3.0	<0.01	3.0	<0.01	3.01	<0.01
s35932	21.6	0.05	16.9	0.19	15.6	1.07	15.6	1.02	18.8	0.01
s38417	19.8	0.04	13.0	0.08	11.9	0.51	11.9	0.49	11.4	<0.01
s38584	16.9	0.03	13.6	1.02	11.6	0.7	11.6	0.66	10.3	0.01
ispd09f31	75.5	0.01	71.0	0.02	71.0	0.02	62.2	0.07	68.7	0.01
ispd09f34	49.1	<0.01	50.8	0.01	50.8	0.01	46.9	0.02	54.9	<0.01

whole system. While it is expected that optimizing the clock tree is expected to reduce a significant amount of the total noise – clock trees consume roughly 30-50% of the total power [1, 2, 3] and the power consumption itself is the source of the noise – we validate this assumption by simulating a full circuit.

	Unoptimized	WaveMin
Clock tree only	Case A	Case B
Full circuit	Case C	Case D

HSPICE simulations were done for the four cases as tabulated above. Benchmark circuit s15850 was synthesized and simulated. In cases A and B, only the clock tree of the circuit was optimized and simulated whereas in cases C and D, the rest of the circuit were simulated also. The input signals to the primary inputs were generated randomly, each having 50% chance of switching at each clock cycle. The fully synthesized circuit had total 385 cells and 4 of them were clock buffers. Note that the power and ground networks were stabilized with decoupling capacitors as described in section 2.6.1.

Fig. 2.8 shows noise waveforms of cases A and B, where A and B are presented as red and blue curves, respectively. $v(n1)$ is the clock signal at a clock sink and $v(nv)$ and $v(ng)$ are voltage fluctuations at the V_{DD} and Gnd lines, respectively. Similarly, Fig. 2.9 shows noise waveforms of cases C and D. Cases A and B are consistent with the experimental results in the previous section and successfully reduced both V_{DD} and Gnd noise. In cases C and D, it is observable that the noise reduction is still valid even when the combinational logics are considered, although the peak-to-peak swing had increased, compared to cases A and D.

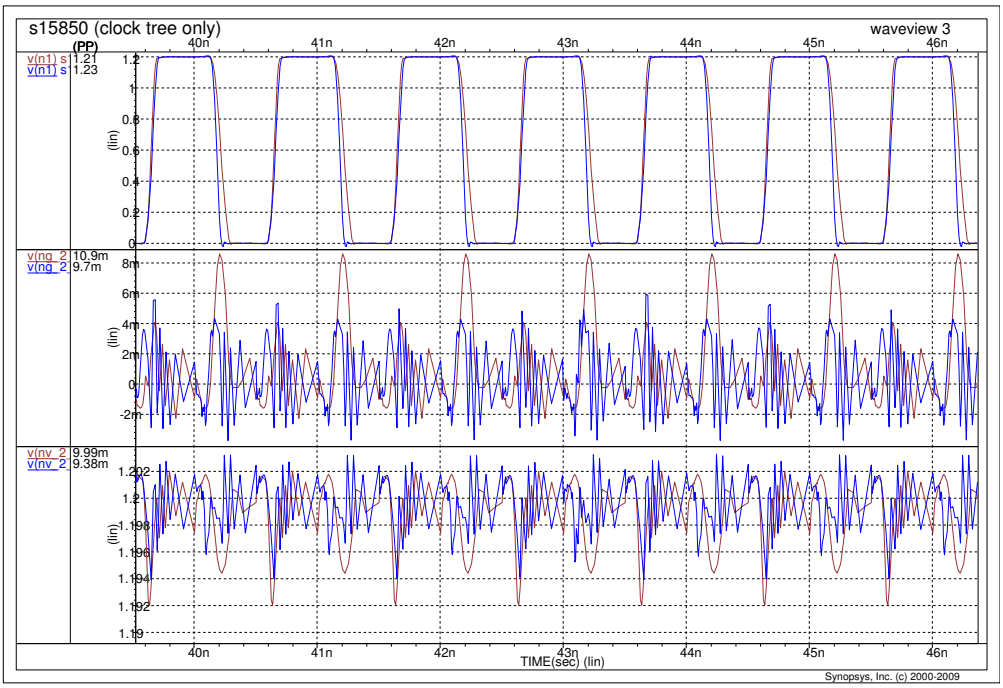


Figure 2.8: Noise waveforms of cases A and B, where A and B are presented as red and blue curves, respectively. $v(n1)$ is the clock signal at a clock sink and $v(nv)$ and $v(ng)$ are voltage fluctuations at the V_{DD} and GND lines, respectively. The peak-to-peak voltage swing is shown in left column labeled as (PP).

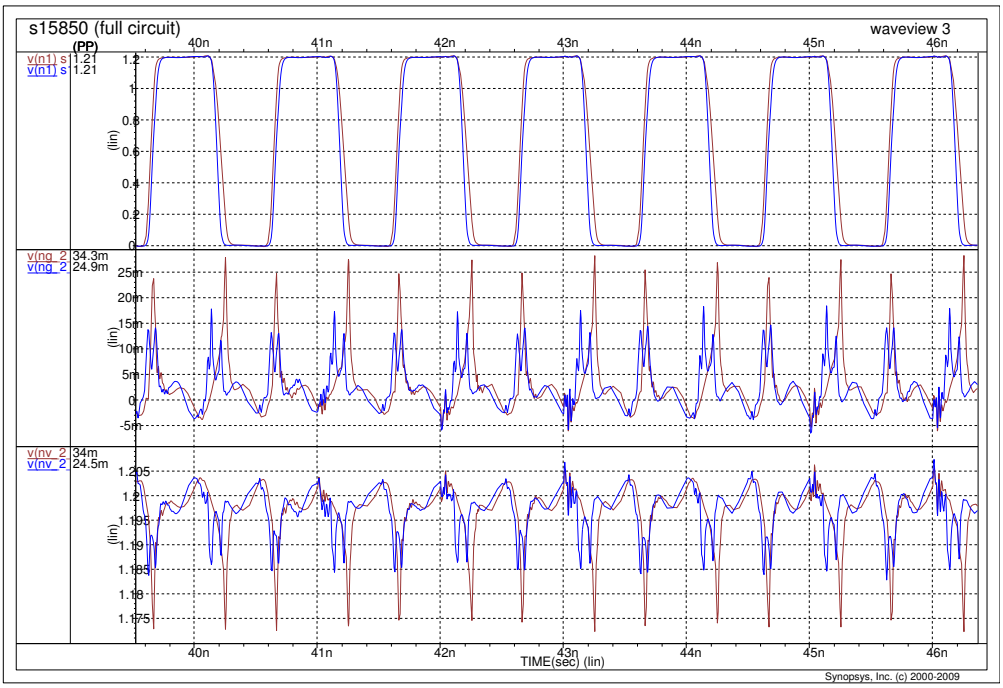


Figure 2.9: Noise waveforms of cases C and D, where C and D are presented as red and blue curves, respectively. $v(n1)$ is the clock signal at a clock sink and $v(nv)$ and $v(ng)$ are voltage fluctuations at the V_{DD} and GND lines, respectively. The peak-to-peak voltage swing is shown in left column labeled as (PP).

2.6 Effects of Clock Polarity Assignment on Simultaneous Switching Noise

Previously, noise reduction was focused on minimizing the peak current emitted by the clock buffers. Although related to the peak current noise, the IR-drop and the ground bounce phenomenon experienced by the circuit are voltage fluctuations, rather than current flow. In this section, we take an in-depth look at the voltage aspect of the noise and propose a method to compensate for the weaknesses of the previously presented optimization methods.

2.6.1 Model of Power Delivery Network

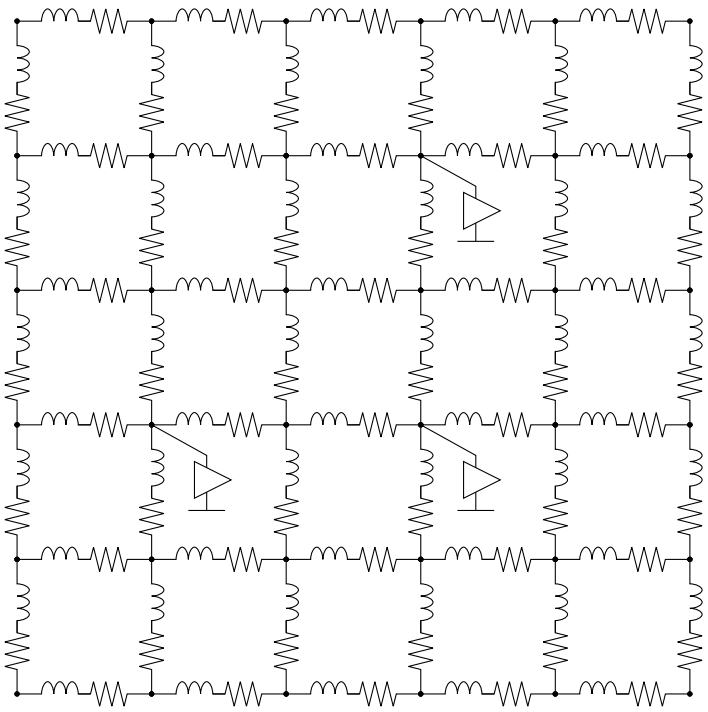


Figure 2.10: Model of Power Delivery Network

Fig. 2.10 illustrates the power delivery network (PDN) model used in the experiments. It is common to model the power networks in high performance

ICs as an RL mesh. In the experiments, each grid cell was $10\mu m \times 10\mu m$ in size. The buffering elements were connected to the closest grid point. The RL parameters are from [21], where $R = 0.007\Omega/\mu m$ and $L = 0.5pH/\mu m$. The input clock signal had 30 ps of slew and frequency of 1GHz. ISCAS'89 benchmark circuits were synthesized using Synopsys Design Compiler and IC Compiler, with Nangate 45nm open cell library [19]. To measure the voltage fluctuations in the PDN, HSPICE simulation was executed on the benchmark circuits.

2.6.2 Peak-to-Peak Voltage Swing

Table 2.3: Peak-to-Peak voltage swing observed in ISCAS'89 benchmark circuits.

Circuit	$\|L\|$	GND noise (mV)			V_{DD} noise (mV)		
		Base	Wavemin	Imp. (%)	Base	Wavemin	Imp. (%)
s13207	8	5.86	2.66	54.56	5.80	2.90	50.00
s15850	3	1.95	4.61	-135.61	2.10	1.70	19.05
s35932	50	39.68	81.86	-106.30	38.50	16.80	56.36
s38417	43	44.92	33.08	26.36	36.20	17.70	51.10
s38584	32	20.55	18.99	7.60	24.40	15.20	37.70
Average				-30.68			42.84

Peak-to-Peak (p-p) voltage swing is a directly observable metric of the noise. Table 2.3 summarizes the p-p voltage swing observed in the ISCAS'89 benchmark circuits. For each junction of the PDN grid, the voltage fluctuations over time was observed and the difference between the maximum and the minimum voltage values was taken as the p-p swing of that junction. The worst (largest) p-p swing values among the junctions was captured as the p-p swing of the circuit.

In all circuits, reductions in V_{DD} noise were achieved. However, s15850 and s35932 experienced GND noise degradation, as shown in Figures 2.11 and 2.12. In these circuits, the one and only buffering element bound to the worst PDN

junction was replaced from BUF_X32 to INV_X16. Although this is a down sizing, the change of polarity increased GND noise while reducing V_{DD} noise. This can be compensated by introducing decoupling capacitors, as will be discussed in the next section.

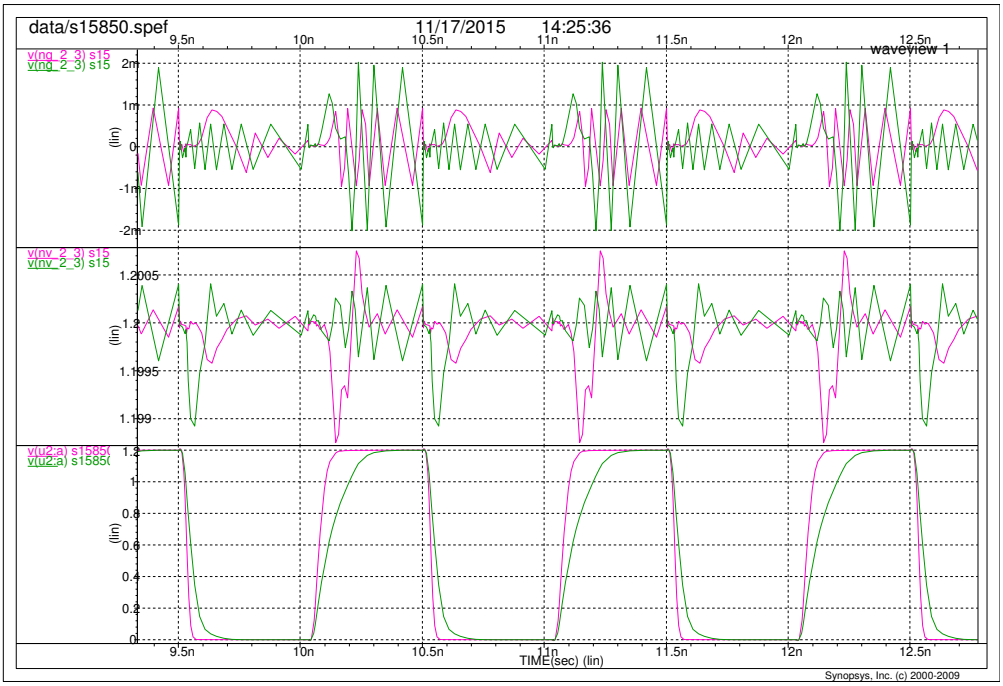


Figure 2.11: Noise waveforms observed in benchmark circuit s15850. The magenta waveforms are from unoptimized input clock tree and the green waveforms are that of the optimized clock tree. The first row is the ground voltage fluctuations over time. The second row shows the power voltage fluctuations. The last row is the input clock signal of the only buffer at the degraded junction.

2.7 Effects of Decoupling Capacitors

In designing high performance chips, decoupling capacitors are an effective method of reducing the noise [22]. Combined with polarity assignment technique, embedding decoupling capacitors can further reduce the noise. Moreover,

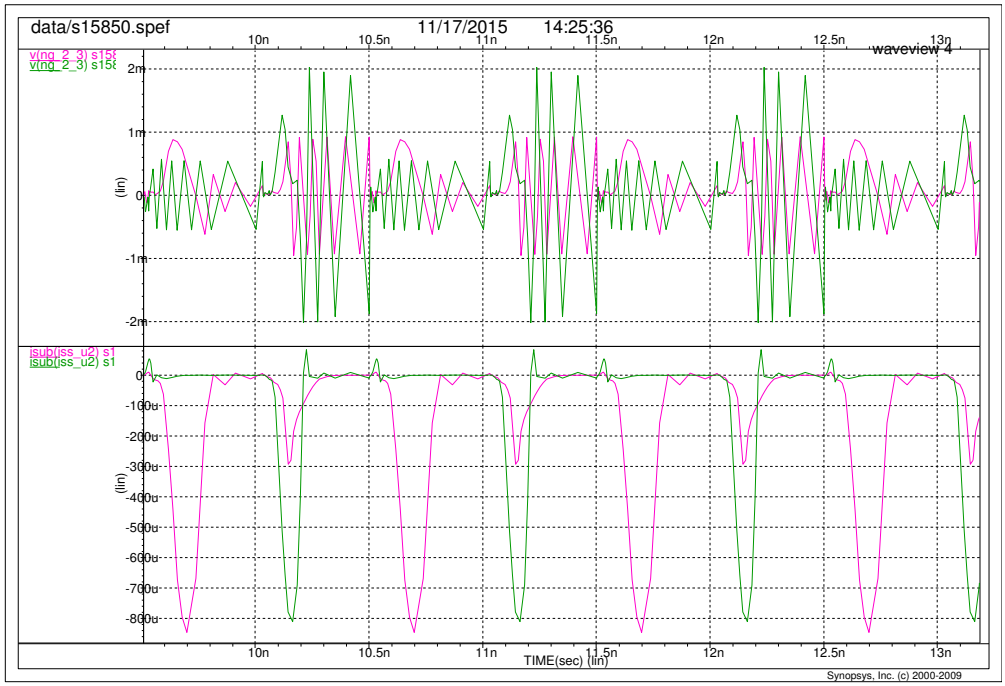


Figure 2.12: Noise waveforms observed in benchmark circuit s15850. The magenta waveforms are from the unoptimized input clock tree and the green waveforms are that of the optimized clock tree. The first row is the ground voltage fluctuations over time and the second row shows the I_{SS} current of the only buffer at the degraded junction. Even though the peak current is smaller for the green waveform, the resulting voltage noise is much larger.

Table 2.4: HSPICE simulation parameters of the PDN in Fig. 2.13

R_p^r, R_p^b, R_p^p	0.025 Ω
L_p^r, L_p^b, L_p^p	0.025 nH
C_b, C_p	50 pF
C_c	30 pF
R_g^r, R_g^b, R_g^p	0.025 Ω
L_g^r, L_g^b, L_g^p	0.025 nH
L_b^C, L_p^C, L_c^C	0 nH
R_b^C, R_p^C, R_c^C	0 Ω

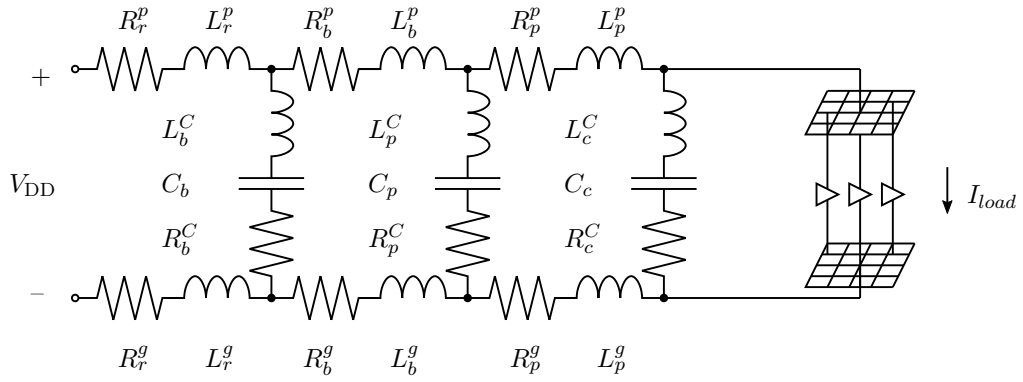


Figure 2.13: Modelling of the decoupling capacitors. Subscripts p and g refer to power and ground paths, respectively. Superscripts r, b, p and c denote voltage regulator, board, package and on-chip PDNs, respectively.

modelling the decoupling capacitors C_b and C_p in the PDN corrects the V_{DD} and GND imbalance observed in the previous section. In this section, we followed the PDN model in [21], as depicted in Fig. 2.13, where the parameters used in the experiments are defined in Table 2.4.

Cases and Notations	Unoptimized	WAVEMIN
No decoupling capacitor	-C-W	-C+W
With decoupling capacitor	+C-W	+C+W

The clock trees were optimized as the four cases tabulated above. Table 2.5 summarizes the results, revealing the effects of decoupling capacitors and WAVEMIN on peak current and V_{DD}/GND noise (C_c = 10 pF). More results with different C_c values are in the Appendix B. Cases with (-C) denote that the on-chip decoupling capacitor C_c is removed. By comparing rows of cases (-C-W) and (-C+W), the effects of the off-chip decoupling capacitors C_b and C_p can be observed: both V_{DD} and GND noise are improved by WAVEMIN, unlike the results in Table 2.3. By comparing rows of cases (-C+W) and (+C-

Table 2.5: The effects of the on-chip decoupling capacitor and WaveMin on noise. $C_c = 10$ pF

Circuit/Case	Peak current (mA)	V_{DD} noise $V_{p-p}(mV)$	Gnd noise $V_{p-p}(mV)$
s13207/-C-W	10.56	27.70	25.65
s13207/-C+W	10.16	25.10	29.32
s13207/+C-W	7.34	33.70	25.02
s13207/+C+W	6.5	27.40	27.01
s15850/-C-W	5.61	14.40	18.09
s15850/-C+W	3.84	23.70	32.60
s15850/+C-W	3.9	17.30	14.52
s15850/+C+W	2.46	24.70	23.64
s35932/-C-W	49.22	120.10	113.52
s35932/-C+W	46.53	106.80	99.33
s35932/+C-W	32.78	120.30	102.45
s35932/+C+W	31.26	105.50	98.94
s38417/-C-W	47.22	109.4	109.18
s38417/-C+W	45.61	106.7	94.37
s38417/+C-W	31.35	111.9	95.57
s38417/+C+W	29.39	104	96.11
s38584/-C-W	39.94	96.6	97.92
s38584/-C+W	38.56	87.4	79.1
s38584/+C-W	26.68	105.7	86.34
s38584/+C+W	25.26	88.4	83.48

W), the effectiveness of WaveMin compared to on-chip decoupling capacitor of 10 pF can be evaluated. In benchmark circuit s38417, they are on par whereas in s38584, WaveMin outperforms the decoupling capacitor. On the other hand, in s15850, the decoupling capacitor has better outcome. It can be said that polarity assignment is roughly equivalent to decoupling capacitor of 10 pF. However, the best case is combining both methods (+C+W), in all circuits. Note that decoupling capacitors come at the cost of capacitor area whereas polarity assignment technique reduce buffer area, as it is shown in Table 3.1. In 45nm technology, the capacitance density is 5-10 fF/μm^2 for MOS capacitors[3]. This implies that WaveMin is roughly equivalent to $1000\mu m^2$ of on-chip area.

2.8 Effects of Clock Polarity Assignment on Clock Jitter

Theoretically, reducing the noise should stabilize the V_{DD} and Gnd voltages, improving the quality of the clock signal. However, with a small voltage drop, the improvement was not measurable. The inductances of power delivery network L_b^C, L_p^C and L_c^C had been increased to 0.1 nH for the observation. Fig. 2.14 shows the jitter histograms from the unoptimized clock tree (left red) and optimized clock tree, by WaveMin (right blue). The optimized clock tree has less standard deviation than the unoptimized clock tree.

2.8.1 Noise in Frequency Domain

The p-p voltage swing was measured for all benchmark circuits while varying the input clock frequency. Fig. 2.15 illustrates the results observed in circuit s15850. The swing appears to be independent of the clock frequency. Running AC analysis through HSPICE reveals that the power distribution network is

[3]Scaled the parameters given in Table 2.1 of [21].

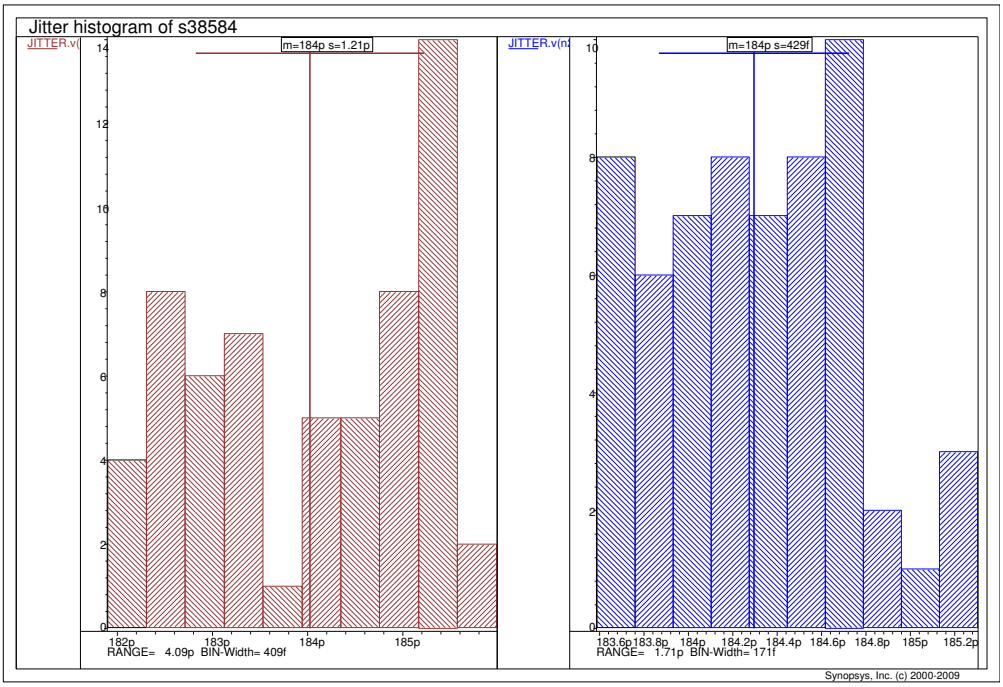
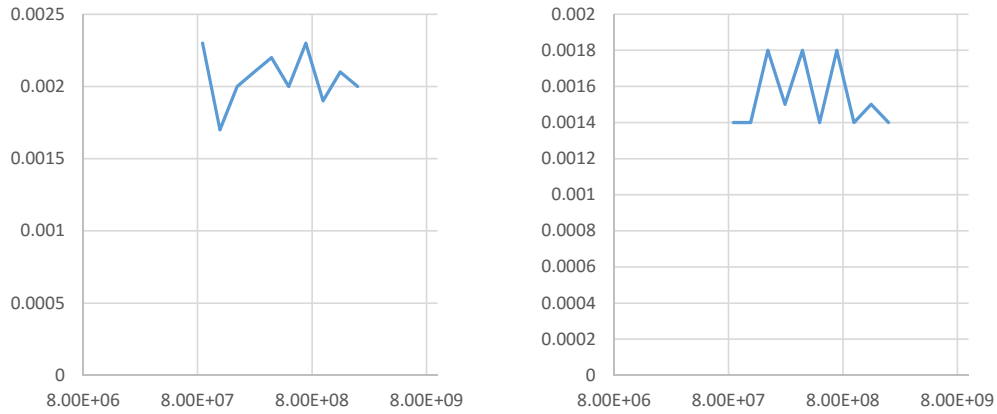


Figure 2.14: Jitter histogram observed in s38584. The red histogram on the left is the jitter histogram of the unoptimized clock tree and the blue histogram on the right is by WaveMin.



(a) Unoptimized (b) WaveMin

Figure 2.15: The frequency response of noise of benchmark circuit s15850, (a) of the unoptimized input clock tree, (b) output clock tree, optimized with WaveMin.

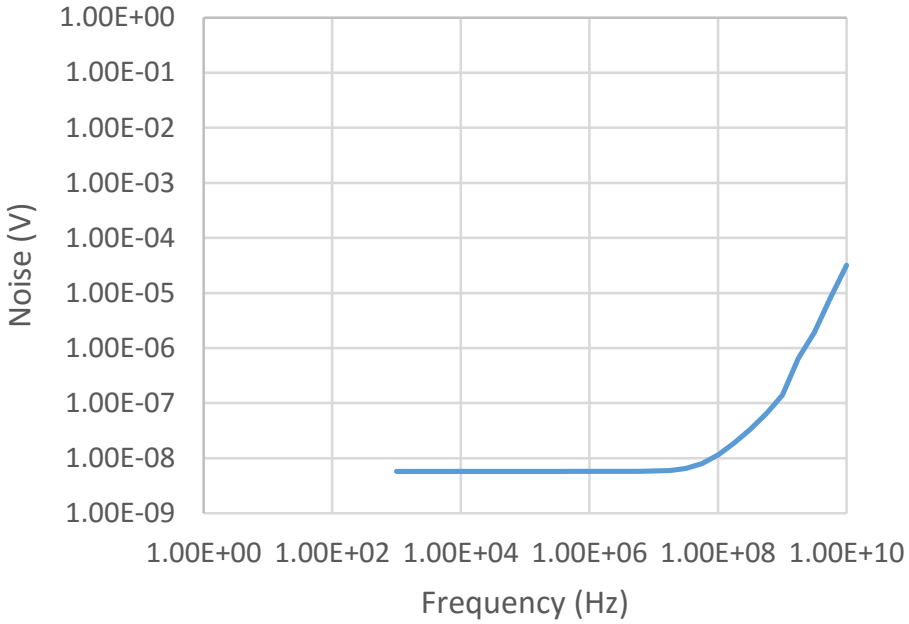
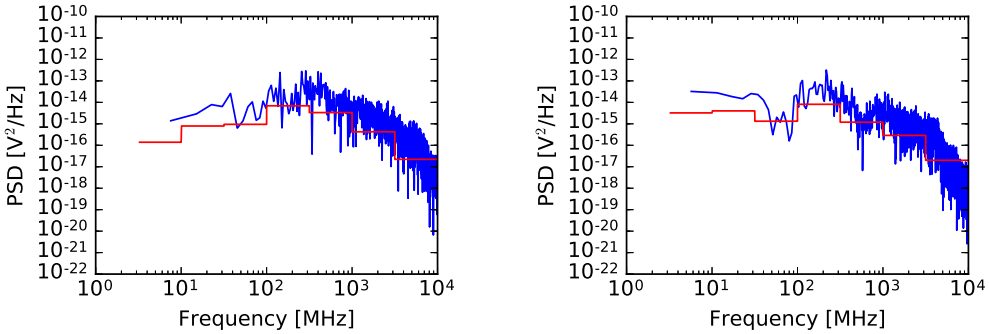


Figure 2.16: HSPICE AC analysis result of clock frequency vs. noise at PDN in circuit s15850

a high-pass filter with the cut-off frequency around 10^8-10^9 Hz (Fig. 2.16). This verifies that, at the frequencies shown in Fig. 2.15, the response should be constant. Altering buffer sizes and the clock polarity does not affect this behavior, as verified by the results in Fig. 2.15(b). These trends also holds in other ISCAS'89 benchmark circuits and similar plots were acquired.

Fig. 2.17 shows the power spectral density of the voltage fluctuations at the center of the PDN mesh, when the frequency of the input signal is 1 GHz. The red bars show the average power in the given frequency band. Both the unoptimized and optimized circuits have the peak powers around 100-1000 MHz band. However, WaveMin tends to reduce power noise at higher frequencies ($>$ 100 MHz). Although the average noise power at lower frequencies have increased, considering that the horizontal axis is in log scale, the overall noise power have decreased. Decoupling capacitor of 30 pF was used to acquire these plots. Power



(a) Unoptimized (b) WaveMin

Figure 2.17: Power spectral density of the supply voltage fluctuations in s15850

spectral density plots of other benchmark circuits are in Appendix A.

2.9 Summary

In this chapter, a comprehensive graph-based algorithm for solving clock polarity assignment problem combined with buffer/inverter sizing, that supports fine-grained peak current noise model was proposed. The experimental results show that the algorithm reduced the peak noise by 15.62% on average, over that by the best known method with coarse-grained noise model [14]. This is attributed to the fact that the fine-grained model allows better exploitation of the clock skew to further reduce the clock noise. Voltage fluctuations on V_{DD} and Gnd was examined to verify that optimizing peak current optimizes voltage noise also.

Chapter 3

Clock Polarity Assignment Under Useful Skew

3.1 Introduction

While there are plenty of research works [23, 24, 11, 25, 12, 13, 14, 26, 27] that addressed the polarity assignment problem, one common feature of all previous works is that they are all *global clock skew bounded* [1] approach. However, for high performance circuits, it is necessary to set a tight clock skew bound since the available time margin is not enough. This means that it becomes much harder to exploit the clock polarity assignment under the tight clock skew bound constraint to minimize the noise. In contrast, the clock polarity assignment under useful skew constraints will be more effective than the global clock skew bound constrained polarity assignment in the sense that it is able to check the setup and hold time constraints between sinks *individually* in the course of the polarity assignment where some sink pairs have loose time margins while some

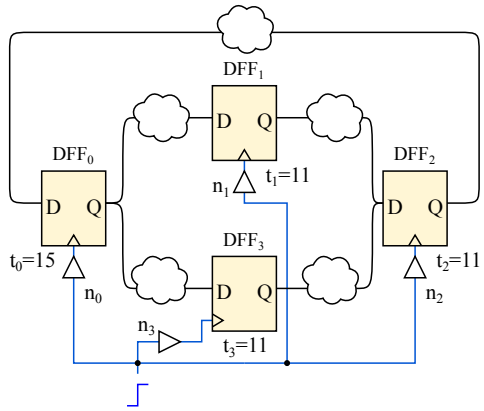
[1] (Global) *clock skew* is defined to the difference between the latest and the earliest clock signal arrival times at the clock sinks.

have tight ones.

The task of determining clock arrival times to every sink is referred to as *useful skew scheduling* [16]. Note that even though there are several works (e.g., [28, 29]) that have utilized the clock skew scheduling to minimize the peak noise, none of them have applied the clock polarity assignment combined with buffer sizing. In this work, we propose a comprehensive solution to the problem of clock polarity assignment integrated with buffer/inverter sizing to reduce clock switching noise. Precisely, (1) we show the polarity assignment problem under useful skew constraints is NP-complete; (2) we propose a clique search based scalable algorithm that is able to trade-off between the solution quality and run time; (3) the proposed algorithm produces library based (practical) solution, so that the optimized buffers and inverters can be taken from the given library.

3.2 Motivational Example

Consider a small clock tree shown in Fig. 3.1(a). It has four clock sinks, each of which has its distinct driving clock buffer. The initial clock signal arrival times to DFF_0 through DFF_3 are 15, 11, 11, and 11, respectively, as indicated by t_0, t_1, t_2, and t_3. Assume that the setup and hold time constraints are pre-calculated and given in Fig. 3.1(b). Given that each of the four clock buffers are initially a buffer instance of type $B1$, we can calculate the arrival time change Δt of its driven FF resulting from replacing it with an instance of other buffer/inverter type, as shown in Fig. 3.1(c). Note that since two clock buffers of the same type in different locations may drive different load capacitances, even when they are to be replaced with another clock buffers of the same type, it is practically required to separately calculate the two values of Δt and their peak power currents at the rising and falling edges of the clock signal (P+ and P-). However, for the sake of simplicity, let us assume that all the clock buffers



(a) Clock design with four clock buffers

-3	$\leq$	$t_0 - t_1$	$\leq$	2
-5	$\leq$	$t_1 - t_2$	$\leq$	4
-3	$\leq$	$t_0 - t_3$	$\leq$	3
-4	$\leq$	$t_3 - t_2$	$\leq$	2
-3	$\leq$	$t_2 - t_0$	$\leq$	2

(b) Setup and hold time constraints

Type	Δt	Noise	
		P+	P-
B1	0	10	3
B2	+2	12	3
I1	0	3	9
I2	+1	3	11

(c) Library of buffers and inverters

#	Type Selection				Skew	Noise	
	n_0	n_1	n_2	n_3		P+	P-
1	B1	B2	B2	B2	2	46	12
2	B1	B2	B2	I2	3	37	20
3	B1	B2	I2	B2	3	37	20
4	**B1**	**B2**	**I2**	**I2**	**3**	**28**	**28**
5	I1	B2	B2	B2	2	39	18
6	I1	B2	B2	I2	3	30	26
7	I1	B2	I2	B2	3	30	26
8	I1	B2	I2	I2	3	21	34

(d) Eight feasible polarity assignment with sizing

Figure 3.1: An illustration of clock buffer polarity assignment problem under useful clock skew condition. (a) The input clock tree, (b) Setup and hold time constraints between the sinks, (c) Buffer and inverter types in the library and (d) The 8 feasible clock polarity assignment of the design (out of $4^4 = 256$ search space) in (a) using the library in (c) that satisfies the time constraints in (b).

in the example have the same Δt, P+ and P- values.

Now, we are ready to perform polarity assignment/buffer sizing. In this example, we resort to brute force method to exhaustively explore the design space. Out of 4^4 combinations, it is found that only eight are *feasible* in that they cause no violation to the constraints given in Fig. 3.1(b). The eight assignments are listed in the table of Fig. 3.1(d). The upper bound values of power/ground noise of an assignment are calculated by summing the P+/P- values given in the library of the assigned types. After the computation of noise upper bounds, the assignment with the minimum worst case ($= 28 = min(max(P+, P-))$) noise is selected as the best assignment, which is assignment #4.

On the other hand, the previous clock polarity assignment and buffer sizing algorithms can only take one clock skew bound for their clock skew specification. Thus, to satisfy every setup and hold time constraint, the designer must select the tightest constraint as the clock skew bound (=2 in this example). Under this tight constraint, only assignments #1 and #5 are feasible, which results in the peak noise of 39, which is 39% higher than that of the useful clock skew optimization result. This example clearly shows that the bounded skew approach may severely limit the exploration of search space and a useful clock skew approach is essential to fully explore the search space in order to find a clock polarity assignment with a minimal peak noise.

3.3 Problem Formulation

Problem 5 (UsefulMin)**.** *Given a buffer library B, an inverter library I, a set L of leaf buffering elements, and a set S of time sampling slots, find a mapping function $\phi : L \mapsto \{B \cup I\}$ that minimizes the quantity of*

$$\max_{s \in S} \left\{ \sum_{e_i \in L} noise(\phi(e_i), s) \right\} \tag{3.1}$$

under a set C of setup and hold time constraints:

$$LB(e_i, e_j) \leq t_i(\phi) - t_j(\phi) \leq UB(e_i, e_j),\ \forall i, j,\ i \neq j$$

where $LB(e_i, e_j)$ and $UB(e_i, e_j)$ can be $-\infty$ and ∞ respectively, if there is no time constraint between e_i and e_j. The term $noise(\phi(e_i), s)$ is the value of peak current estimation at a time sampling slot s caused by the switching of e_i when it is assigned with $\phi(e_i) \in \{B \cup I\}$.

Note that P+ and P- values in the motivational example are short names for $noise(\phi, s_1)$ and $noise(\phi, s_2)$, respectively when $|S| = 2$, in which the peak current noise sampling slots s_1 and s_2 will be used as the high/low periods in the clock cycle. Increasing the number of time sampling slots can improve noise estimation. Moreover, both I_{DD} and I_{SS} should be sampled as slots since the objective is to minimize the worst current.

UsefulMin is an intractable problem, since UsefulMin is a more general problem than WaveMin problem in Chapter 2. The formal proof are as follows. By Theorem 1, the decision version of WaveMin problem, decision-WaveMin (Problem 2) is NP-Complete. The decision version of UsefulMin problem is defined as follows:

Problem 6 (decision-UsefulMin)**.** For a UsefulMin instance with (L, B, I, S, C) and a constant z, is there a mapping ϕ such that the value of (3.1) is less than or equal to z?

Theorem 2. decision-UsefulMin is NP-complete.

Proof. decision-UsefulMin is in NP since for a mapping result, (3.1) can be computed in polynomial time. decision-UsefulMin is in NP-hard since any instance of decision-WaveMin problem can be reduced to an instance of decision-UsefulMin problem by converting the clock skew bound κ into the set C of constraints for each pair of leaves, which can be done in $O(|L|^2)$-time. The solution instance obtained by solving the decision-UsefulMin instance is directly compatible with decision-WaveMin. □

3.4 Proposed Algorithm

3.4.1 Integer Linear Programming Formulation and Linear Programming Relaxation

While it is possible to formulate UsefulMin problem into 0-1 Integer Linear Programming (ILP), the task of exploring feasible solutions can be transformed into a variant of maximum clique search problem, as will be shown later. In this case, Linear Programming (LP) relaxation heuristic is of little help; it is known that the LP relaxation of unweighted maximum clique problem – weight in UsefulMin problem being noise – yields poor solutions: the optimal solution of LP relaxation has one of 0, 1, and 1/2 for each variable, which in most cases only few of the variables have integer values. This makes the gap between the optimal 0-1 ILP solution and the relaxed LP solution too large [30].

3.4.2 Formulating into Maximum Clique Problem

Consider the UsefulMin problem instance of the clock tree in Fig. 3.1, which is then represented by a weighted graph $G(V, E, W)$ as shown in Fig. 3.2: (i) for each pair $(n_i, B_j/I_j)$ of leaf buffers $n_i \in L$ and buffers/inverters $B_j/I_j \in B \cup I$, there is a unique vertex in V, and $|V| = |L| \times (|B| + |I|)$; (ii) there exists an edge in E between vertices $(n_i, B_j/I_j)$ and $(n_k, B_l/I_l)$, $i \neq k$, if and only if assigning n_i with B_j/I_j and n_k with B_l/I_l causes no setup and hold time violation. For example, there is no edge between (n_0, I_1) and (n_1, I_2) in Fig. 3.2 since a precise analysis leads to find the violation of $-3 \leq t_0 - t_1 \leq 2$ in Fig. 3.1(b). Note that the vertices in the same row in Fig. 3.2 have no edge between them. This forbids a leaf buffer to be assigned to more than one type of buffer/inverter. In addition, there will be edges between all possible pairs of nodes in rows marked n_1 and n_3 since there is no skew constraint at all in the initial clock tree between the sinks corresponding to n_1 and n_3; (iii) weight $w_i \in W$ assigned to a node $(n_i, B_j/I_j)$

represents the set of power/ground currents at the sampling slots in S when B_j or I_j assigned to n_i switches. Let $w_i(s_j)$ be the power/ground current at sampling slot s_j when the buffer/inverter assigned to leaf buffer e_i switches. Then, the problem of finding the clock polarity assignment under the useful skew constraints is equivalent to the problem of finding a clique $Q \subset V$ of size $|L|$ in G that minimizes the (noise) quantity of

$$max\{\sum_{q_i \in Q} w_i(s_j), j = 1, \cdots, |S|\}. \tag{3.2}$$

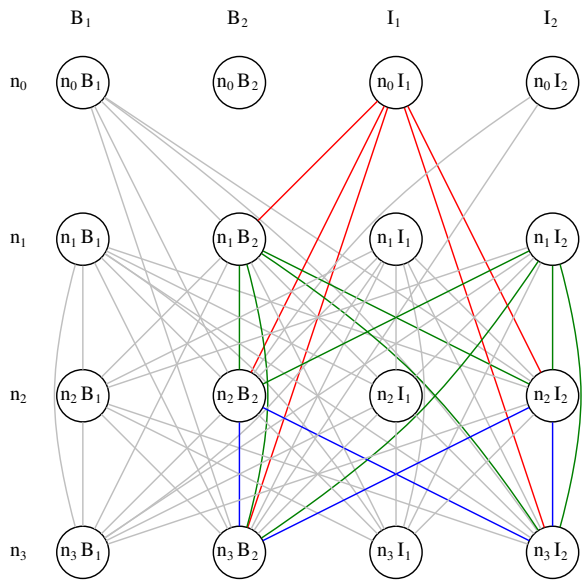


Figure 3.2: Transformation of the problem instance in Fig. 3.1 into a search problem in a graph $G(V, E, W)$.

Since there is no edge between the vertices in the rows of G in Fig. 3.2, the problem of finding $|L|$-clique with the minimum value of (3.2) can be translated to finding a maximum clique in G with the minimum value of (3.2). Thus, if the size of maximal clique in G is less than $|L|$, there is no feasible polarity assignment that meets all useful skew constraints. For example, in Fig. 3.2, there

are eight cliques of size 4 that can be found from the subgraph defined by vertices $\{(n_0, B_1), (n_0, I_1)\}$, (n_1, B_2), $\{(n_2, B_2), (n_2, I_2)\}$, and $\{(n_3, B_2), (n_3, I_2)\}$, which correspond to the eight feasible assignments in Fig. 3.1(d). Among the assignments, assignment #4 produces the least value of (3.2).

3.4.3 Scalable Algorithm for Clique Exploration

The problem of finding a maximum clique with least cost is known to be not only intractable but also hard to approximate [30]. Hence, we propose to employ local search heuristics to find a good feasible solution, as summarized in Fig. 3.3.

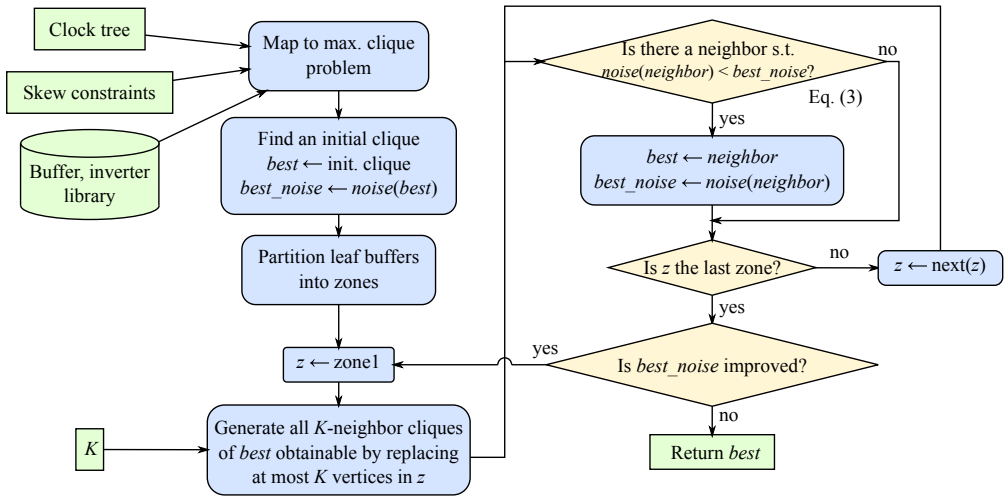


Figure 3.3: The flow of UsefulMin algorithm.

We start by mapping the UsefulMin problem instance to a maximum clique problem instance. To use local search heuristic, we first need to find an initial clique of cardinality $|L|$ to start the local search. A trivial solution is the unoptimized one, where no buffers are changed. However, note that, since the initial clique for the local search determines the quality of the final solution, it is desirable to use a previous skew bound constrained clock polarity assignment/buffer sizing algorithm to find an initial clique. Then, we iteratively search

for cliques that yield better results. We search them by finding K-neighbors of the clique found in the current iteration. Clique X is called a K-neighbor of Y if X can be formed by replacing K or less vertices of Y. Since the designer is able to control the value of parameter K, it is possible to trade-off between the solution quality and run time.

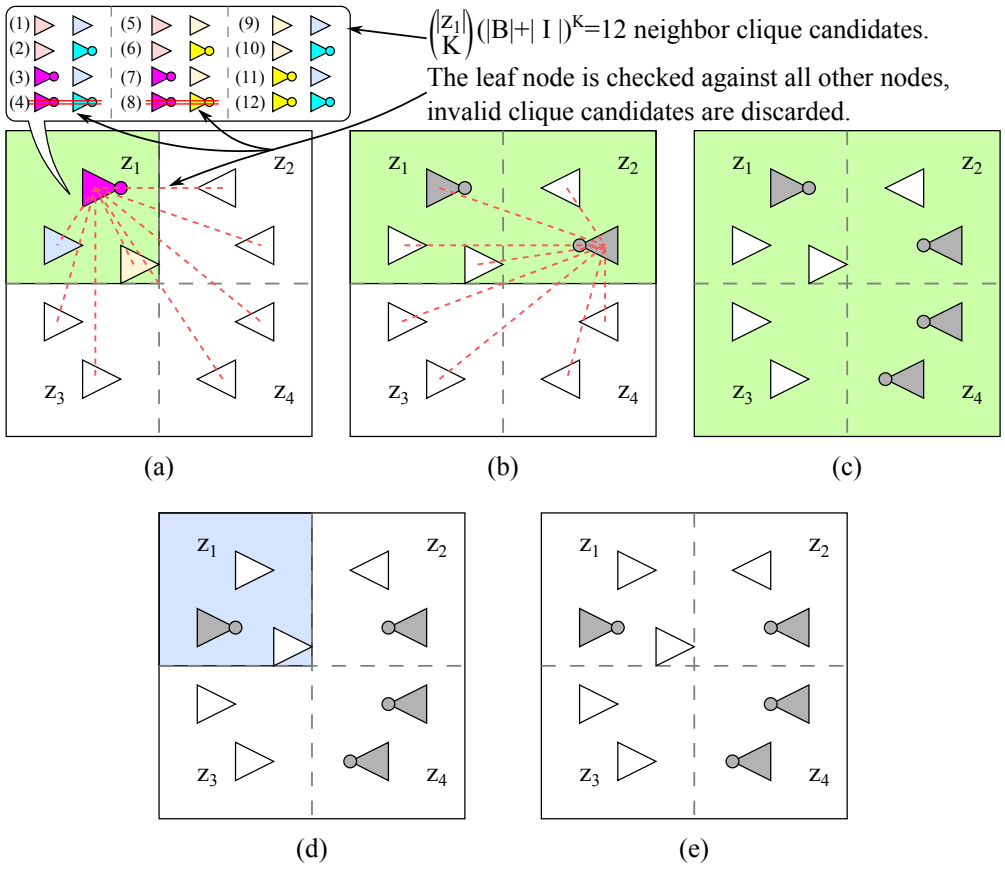


Figure 3.4: An example of illustrating the procedure of UsefulMin algorithm. (a) The first zone z_1 is optimized. Each of the 2-neighbor clique candidates is checked if it forms a clique globally. (b) Among the candidates, the one with the least value of (3.2) is frozen and the optimization continues to z_2. (c) All zones are optimized. The optimization restarts from z_1. (d) A better assignment in z_1 has been discovered. (e) UsefulMin terminates when no improvement is made.

Fig. 3.4 shows an example execution of UsefulMin algorithm. Assume that

$K = 2$ and $|B| + |I| = 2$. The leaf buffering elements are partitioned into zones by their locations. In Fig. 3.4(a), zone z_1 is optimized. Since there are $|z_1|(= 3)$ leaf buffering elements in the zone and $K = 2$, $\binom{|z_1|}{k}(|B|+|I|)^k = 12$ 2-neighbor clique candidates are generated from this zone. Each candidate is then checked if it globally forms a clique. Among the candidates that form cliques, the one with the least value of noise ((3.2)) is frozen and the optimization continues to the next zone, as shown in Fig. 3.4(b). This process continues until there are no more zones to optimize, as shown in Fig. 3.4(c). Since the new clique in Fig. 3.4(c) has new neighbor cliques, the zone-by-zone optimization is repeated, subsequently generating results in Fig. 3.4(d) and Fig. 3.4(e).

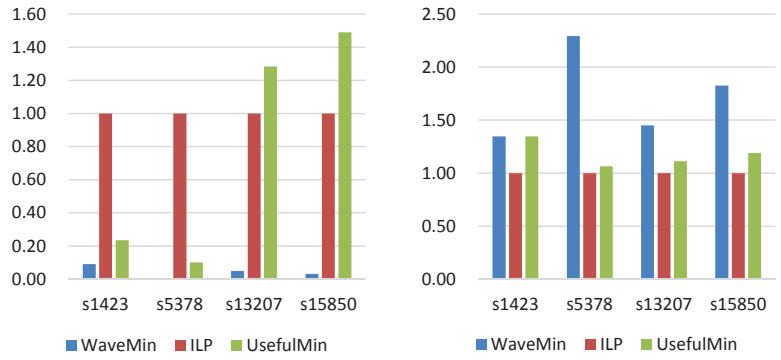
Theoretically, raising K increases the search space significantly since the number of candidates to be examined is $O(\binom{|L|}{K}(|B| + |I|)^K)$. However, by reflecting the fact that noise is a local phenomenon, we can partition the leaf buffers in L into zones by their proximity and perform the optimization in zone-by-zone manner, which greatly reduces the search space: for each zone, we find K-neighbor cliques where the K vertices are only chosen from the zone. From the K-neighbors, we keep the neighbor clique with the least *noise* as the new best clique and move on to the next zone. When all zones are visited, we start the search again from the first zone, as the new best clique may have better neighbor cliques. This exploration is repeated until no improvement is made.

The run time analysis of the zone based algorithm is as follows. Suppose there are $|Z|$ zones. Then, there are $n = |L|/|Z|$ leaf nodes in each zone on average. Since there are $O(\binom{n}{K}(|B| + |I|)^K)$ K-neighbor candidates for each zone, $O(n^{K-1}(|B|+|I|)^K|L|)$ K-neighbors are searched in the whole circuit. The overall runtime of a single iteration is $O(n^{K-1}(|B| + |I|)^K|L|) \times (O(K|L|) + O(2|S|))$, where $O(K|L|)$ time is used for checking if the new set of vertices

form a clique and $O(2|S|)$ is for incrementally computing noise. Simplifying the expression yields $O(Kn^{K-1}(|B|+|I|)^K|L|^2)$, assuming $|S|$ is much smaller than $K|L|$. Although setting $K = 1$ greatly reduces execution time, it is desirable to have a reasonable value so that runtime and solution quality can be traded off reasonably.

3.5 Experimental Results

3.5.1 Experimental Setup



(a) Execution time (b) Peak current

Figure 3.5: Normalized comparison of UsefulMin, conventional WaveMin, and optimal ILP formulation. ISCAS'89 benchmarks were used since ISPD'10 benchmarks were too large for the ILP solver. In small circuits, the heuristic iteration can take longer than ILP (0.18s vs. 0.25s for s15850). However, as circuits become larger, ILP overtakes (3.9s vs. 0.8s in s5378, where s5378 is the largest of the 4 benchmark circuits).

The proposed algorithm UsefulMin was implemented in C++ language on a Linux machine. Clock trees were generated for ISPD'10 high performance clock network synthesis contest benchmarks with the algorithm in [20], using Nangate 45nm Open Cell Library [19] and employing only BUF_X8 as buffering elements. Since ISPD'10 benchmarks have only clock sink information and no

Table 3.1: The comparison of the useful skew constrained polarity assignment (UsefulMin) and skew bound constrained assignment (WaveMin). The useful skew constraints were within [60, 90] ps for UsefulMin. # Constr. and # Lvs columns indicate the number of useful clock skew constraint and the number of leaf buffering elements in the clock tree, respectively. WaveMin requires tight skew bound to meet the skew constraints, which is 60 ps, while UsefulMin exploits individual skew constraints during the polarity assignment, lowering down the peak noise by 10.9% further. *Area* indicates the total area of leaf buffers/inverters allocated by the polarity assignment.

ISPD 2010 Benchmark Circuits	# Constr.	# Leaves	Base		Wavemin [31]		Ours		Improvement	
			Noise (mA)	Area (μm^2)	Noise (mA)	Area (μm^2)	Noise (mA)	Area (μm^2)	Noise (%)	Area (%)
01	11070	221	235.1	22.38	155.3	21.24	122.1	13.73	21.38	35.38
02	22490	454	433.9	45.97	281.9	43.83	242.4	29.62	14.01	32.42
03	12000	113	106.1	11.44	45.58	6.00	42.56	6.49	6.63	-8.26
04	18450	116	115.3	11.75	52.85	8.58	48.21	6.60	8.78	23.07
05	10160	49	51.83	4.96	35.12	4.84	27.78	3.09	20.90	36.05
06	9810	77	74.26	7.80	43	6.28	40.25	4.88	6.40	22.22
07	19150	131	125.4	13.26	73.26	13.20	67.13	8.51	8.37	35.55
08	11340	89	90.88	9.01	51.43	7.05375	51.05	5.88	0.74	16.59
Average									**10.90**	**24.13**

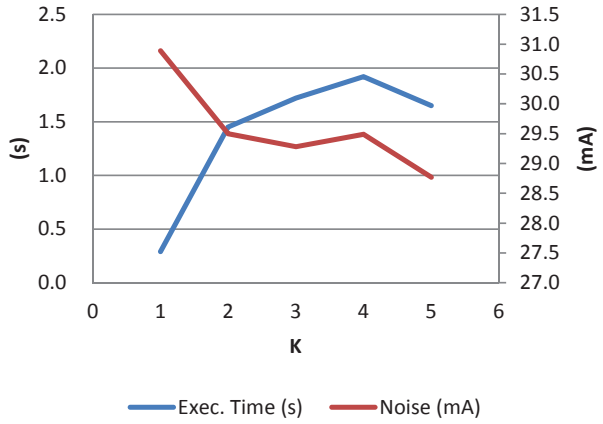


Figure 3.6: The effect of parameter K on the optimization of circuit 05 in ISPD'10 by UsefulMin algorithm.

circuit/individual clock skew constraint information, the setup and hold time constraints were generated randomly within [60, 90] ps range for upper bounds and [-90, -60] for lower bounds. To compare the results with that of a skew bound constrained clock polarity assignment/buffer sizing approach, WaveMin was selected. We set buffer library $B = \{BUF\_X4, BUF\_X8, BUF\_X16\}$ and inverter library $I = \{INV\_X4, INV\_X8, INV\_X16\}$. Leaf buffering elements were partitioned into zones by their locations, recursively bisecting each zone until every zone had 10 or less leaf buffers. After polarity assignment, HSPICE simulation was run on the clock trees to measure the peak noise current.

3.5.2 Assessing the Performance of UsefulMin over Wavemin

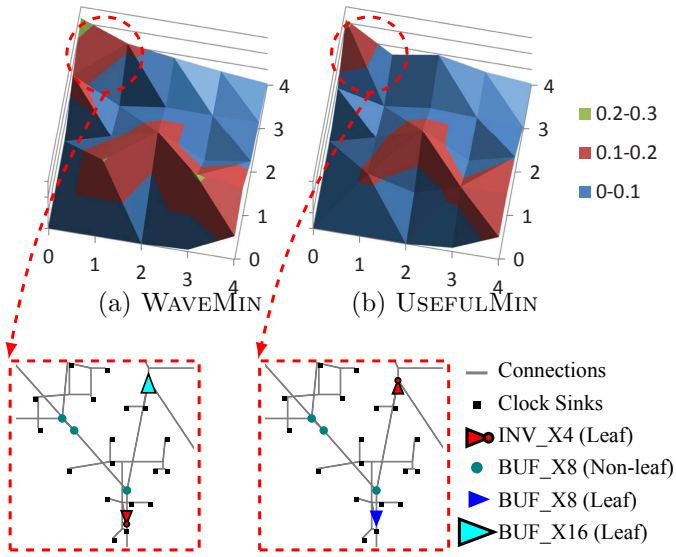
The simulation results are summarized in Table 3.1. # Constr. and skew range columns show the information on clock constraint generation. For each benchmark, clock skew constraints were randomly generated so that the number of constraints is equal to 10 times the number of clock sinks, where the absolute value of upper and lower bound of the constraints are given as the skew range column. Since WaveMin is a clock skew bounded algorithm, it was run with

the tightest clock skew bound (=60 ps). UsefulMin used the solution from WaveMin as its initial clique and searched neighbor cliques with $K = 5$. Overall, the algorithm reduces the peak noise by 49.1% and 10.9% further on average over that of no polarity assignment and the conventional polarity assignment, respectively. On average UsefulMin reduces the power by 4.9% over that of WaveMin. The minimum and maximum average power improvement are -2.5% in circuit 03 and 12.5% in circuit 02, respectively, which reveal similar trend to that of the area improvement. Fig. 3.5 compares UsefulMin algorithm with optimal ILP formulation. For the ILP solver, SCIP [32] was used. The two curves in Fig. 3.6 show how UsefulMin algorithm trades the noise value with the run time as the setting of parameter K changes in the module of K-neighbor clique search. It reveals that UsefulMin algorithm can effectively control the noise quality while taking into account the execution time.

Finally, Fig. 3.7 shows the geometric distributions of the voltage fluctuation in circuit 07 in ISPD'10 optimized by WaveMin and UsefulMin. The comparison shows that by carefully spreading buffers and inverters while meeting all local skew constraints, UsefulMin reduces the regional noises more effectively than the other.

3.6 Summary

In this chapter, a scalable solution to the problem of the clock polarity assignment under useful clock skew constraints is proposed. Unlike the conventional (global) clock skew bound constrained approaches, the new method exploited individual clock skew constraints to further reduce the peak current. Precisely, we formulated the problem into the maximal clique exploration problem and employed a K-neighbor search scheme to trade-off the run time and quality of polarity assignment. For designing high speed systems with tight time margins,



(a) WaveMin (b) UsefulMin

(c) Clock tree, optimized with WaveMin (d) Clock tree, optimized with UsefulMin

Figure 3.7: Geometric distribution of voltage fluctuation in circuit 07 in ISPD'10. Units are in Volts. (a) WaveMin optimized the voltage drops successfully. (b) UsefulMin optimized the noise further by exploiting useful skews. Subfigures (c) and (d) show a small section of the clock tree near grid (0, 4). Initially, all of the buffering elements are BUF_X8. Then, (c) WaveMin replaces many leaf nodes with buffers and inverters of different sizes for noise reduction. (d) UsefulMin discovers that it is possible to further reduce noise by removing BUF_X16 (cyan triangle) in (c) and allocating BUF_X8 (blue triangle).

the proposed approach would be useful in mitigating the clock noise, which otherwise the conventional polarity assignment approaches could rarely achieve.

Chapter 4

Extensions of Clock Polarity Assignment Methods

4.1 Coping With Thermal Variations

4.1.1 Introduction

The non-uniform temperatures on a chip as well as the significant on-chip thermal gradient which are occurring during the execution of chip circuits of high power density are the main cause of the high delay variations [33]. Since the clock nets are one of the most sensitive signals to the delay variations caused by the thermal variation [34, 35], it is important to consider the effect of thermal variation on the polarity assignment in the clock tree synthesis. There are a couple of works which have considered the clock tree synthesis under the thermal variation. TACO [36] constructed a tree that balances the clock skew under the two given static thermal profiles, one uniform and the other worst. The reason of choosing only the two thermal profiles is that analyzing and optimizing all the transient thermal profiles between the two profiles is an extremely difficult

task. Burito [37] then extended the Taco's work to the clock tree synthesis in the 3D IC designs.

4.1.2 Proposed Method

In this section, we extend WaveMin as WaveMin-t to cope with thermal variations, by adjusting the clock signal arrival times through buffer/inverter sizing. The major difference between our work and the works in Taco and Burito is that the task of Taco and Burito is to restructure the initial clock tree routing with the objective of minimizing the additional clock wirelength while balancing and minimizing the clock skew of the worst thermal profile, whereas the task of WaveMin-t is to determine the buffer sizing and polarity assignment with the objective of minimizing the power/ground noise while satisfying the clock skew constraint under the thermal variation. That is, WaveMin-t preserves the routing of the initial clock tree.

Let us suppose that we are given M chip thermal profiles P_1, P_2, $\cdots$, and P_M which are extracted during the execution of the circuits in the chip, where we assume P_1 is the uniform (lowest) temperature profile of the circuit just before the execution. Then, the thermal profiles may produce different clock skews on the same clock tree, causing clock skew variation. This means that our thermal aware polarity assignment and buffer sizing requires to satisfy the clock skew constraint under every thermal profile. (Note that the value of peak current, for the same buffer sizing and polarity assignment, may go down as the chip temperature goes up due to the increase of the delay.) However, since the places in which the buffer sizing and polarity assignment are confined to the relatively small and short-distance regions that contain the sink buffering elements, we assume that the peak current for a solution of polarity assignment

and buffer sizing is invariant with respect to the temperature.[1] The problem we want to solve for a sub-area on a chip can be stated as:

Problem 7 (Thermal aware polarity assignment and buffer / inverter sizing for noise minimization)**.** *For a sub-area that contains a set L of sink buffering elements, a buffer type set B, an inverter type set I, thermal profiles P_1, P_2, $\cdots$, and P_M, and clock skew bound κ, find a mapping function $\phi : L \mapsto \{B \cup I\}$ that minimizes the quantity of*

$$\max_{s \in S} \left\{ \sum_{e_i \in L} noise(\phi(e_i), s) \right\} \tag{4.1}$$

$$s.t.\ t_{skew,j} \leq \kappa, \forall j = 1, \cdots, M$$

where $t_{skew,j} = \max_{i=1,\cdots,|L|}(arr\_max_j(\phi(e_i)))$ - $\min_{i=1,\cdots,|L|}(arr\_min_j(\phi(e_i)))$, in which $arr\_max_j(\phi(e_i))$ and $arr\_min_j(\phi(e_i))$ represent the latest arrival time and the earliest arrival time from the clock source to FFs that are connected directly to e_i under thermal profile P_j, respectively and $noise(\phi(e_i, s))$ indicates the amount of peak current on $\phi(e_i)$ under P_1 at time sampling slot s.

We solve the problem of satisfying all the clock skew constraint under all thermal profiles P_1, P_2, $\cdots$, and P_M by manipulating feasible time intervals as follows: For each P_j, we generate all the feasible time intervals. Let us denote the set of feasible time intervals corresponding to P_j by $\mathcal{H}_j$ $= \{H(t_{(j,1)}), H(t_{(j,2)}), \cdots\}$. In addition, let $C_{(j,k)}(e_i)$ denote the set of buffers and inverters such that the values of $arr\_max(\cdot)$ and $arr\_min(\cdot)$ for their assignments to sink $e_i \in L$ are in $H(t_{(j,k)})$ in set $\mathcal{H}_j$. Then, by the definition of feasible time intervals, $C_{(j,\cdot)}(e_i) \neq \emptyset$, for each sink $e_i \in L$ and time interval $H(t_{(j,\cdot)}) \in \mathcal{H}_j$. Each feasible time interval is characterized by its $C_{(j,\cdot)}(e_i)$'s.

Definition 1 (Intersection of feasible time interval sets)**.** *The intersection, denoted as $\mathcal{H}_{(j,l)}$, of two feasible interval sets $\mathcal{H}_j$ and $\mathcal{H}_l$ is defined as the set of the intersection, denoted as $H(t_{(j,l,\cdot)})$, of every pair of their elements $H(t_{(j,\cdot)}) \in$*

[1] In our work, the peak current value under the initial thermal profile P_1 is used as the representative value of peak currents over all thermal profiles, which can in fact be used as an upper bound of the peak currents under P_2, $\cdots$, P_M because the temperature on P_1 is the lowest.

$\mathcal{H}_j$ and $H(t_{(l,\cdot)}) \in \mathcal{H}_l$, (The intersection of two feasible time intervals $H(t_{(j,\cdot)})$ and $H(t_{(l,\cdot)})$ is characterized by the set intersection $C_{(j,\cdot)}(e_i) \cap C_{(l,\cdot)}(e_i)$ for every sink $e_i \in L$.), and satisfying that $H(t_{(j,l,\cdot)})$ is a feasible time interval. ($H(t_{(j,l,\cdot)})$ is called a feasible time interval for $H(t_{(j,\cdot)})$ and $H(t_{(l,\cdot)})$ of thermal profiles P_j and P_l if $C_{(j,\cdot)}(e_i) \cap C_{(l,\cdot)}(e_i) \neq \emptyset$ for every $e_i \in L$.)

WaveMin-t will compute the intersection of all feasible time interval sets of P_1, P_2, $\cdots$, P_M incrementally: $\mathcal{H}_{(1,2)}$ is obtained from $\mathcal{H}_1$ and $\mathcal{H}_2$. $\mathcal{H}_{(1,2)}$ is then intersected with $\mathcal{H}_3$ to produce $\mathcal{H}_{(1,2,3)}$. This process is repeated until the intersection produces an empty set or $\mathcal{H}_{(1,2,3,\cdots,M)}$ is produced. The generation of empty set in the process of intersection means that there is no feasible time interval which satisfies the clock skew constraint under all thermal profiles P_1, P_2, $\cdots$, P_M. In that case, it may be needed to relax the clock skew constraint by increasing the value of κ and repeat the intersection operation. The next step is then to convert the problem into MOSP problem with the feasible time intervals in $\mathcal{H}_{(1,2,3,\cdots,M)}$.

The example in Fig. 4.1 illustrates the intersection of feasible time intervals. Suppose we have extracted three thermal profiles P_1, P_2, and P_3 where it is assumed that they are the thermal instances at the beginning, in the middle, and at the end of the execution of chip circuit, respectively. Further, suppose that there are five sinks e_1, $\cdots$, e_5, three types of buffer b_1, b_2, b_3, and three types of inverter i_1, i_2, i_3. Fig. 4.1(a) shows an example of the sets $\mathcal{H}_1$, $\mathcal{H}_2$, and $\mathcal{H}_3$, of feasible time intervals produced by feasible interval generation phase of WaveMin for the clock trees of P_1, P_2, and P_3 under the same clock skew bound.[2] Fig. 4.1(b) then shows the result of $\mathcal{H}_1 \cap \mathcal{H}_2$, which is $\mathcal{H}_{(1,2)}$, where two time intervals are feasible. Then, by intersecting each of the two feasible time intervals with that in $\mathcal{H}_3$, we produce the two time intervals as shown

[2]We can see that as the thermal profile changes by the execution of circuit, the number of candidate buffers and inverters on the feasible time intervals is reduced. This is because of the increase of clock delay variation.

	Profile P_1		Profile P_2		Profile P_3
	$H_1(43)$	$H_1(44)$	$H_2(59)$	$H_2(53)$	$H_3(74.3)$
e_1	b_1 i_1	b_1	 i_1	 i_1	b_1, b_2 i_1
e_2	b_1, b_2, b_3 i_1, i_2	b_2, b_3, b_4 i_2, i_3	b_2, b_3 i_1, i_2, i_3	b_3, b_4 i_3	b_2, b_3 i_1
e_3	b_2, b_3 i_1, i_2, i_3	b_3, b_4 i_2, i_3	b_3, b_4 i_2, i_3	b_3, b_4 i_3	b_3 i_2
e_4	b_1, b_2, b_3 i_2, i_3, i_4	b_2, b_3 i_3, i_4	b_1, b_2 i_3, i_4	b_2 i_3	b_1
e_5	b_1, b_2 i_1	b_1, b_2 i_1	b_1, b_2	b_1, b_2 i_1	b_1, b_2 i_4

(a) $\mathcal{H}_1$, $\mathcal{H}_2$, and $\mathcal{H}_3$.

$\mathcal{H}_{(1,2)}$	$H_{(1,2)}(43,59)$	$H_{(.)}(43,53)$	$H_{(.)}(43,59)$	$H_{(.)}(43,53)$
e_1	 i_1	 i_1	$\emptyset$	$\emptyset$
e_2	b_2, b_3 i_1, i_2	b_3	b_2, b_3 i_2, i_3	b_3, b_4 i_3
e_3	b_3 i_2, i_3	b_3 i_3	b_3, b_4 i_2, i_3	b_3, b_4 i_3
e_4	b_1, b_2 i_3, i_4	b_2 i_3	b_2 i_3, i_4	b_2 i_3
e_5	b_1, b_2	b_1, b_2 i_1	b_1, b_2	b_1, b_2 i_1
	feasible	**feasible**		

(b) $\mathcal{H}_{(1,2)}$ produced from (a).

$\mathcal{H}_{(1,2,3)}$	$H_{(1,2,3)}(43,59,74.3)$	$H_{(1,2,3)}(43,59,72)$
e_1	 i_1	i_1
e_2	b_2, b_3 i_1	b_3
e_3	b_3 i_2	b_3
e_4	b_1	$\emptyset$
e_5	b_1, b_2	b_1, b_2
	feasible	

(c) $\mathcal{H}_{(1,2,3)}$ produced from (b) and (a).

Figure 4.1: An example illustrating the derivation of feasible time intervals under multiple thermal profiles.

in Fig. 4.1(c), in which the first one is feasible. Finally, the MOSP phase of WaveMin will be applied to the feasible interval.

WaveMin-t: Thermal aware polarity assignment and buffer sizing

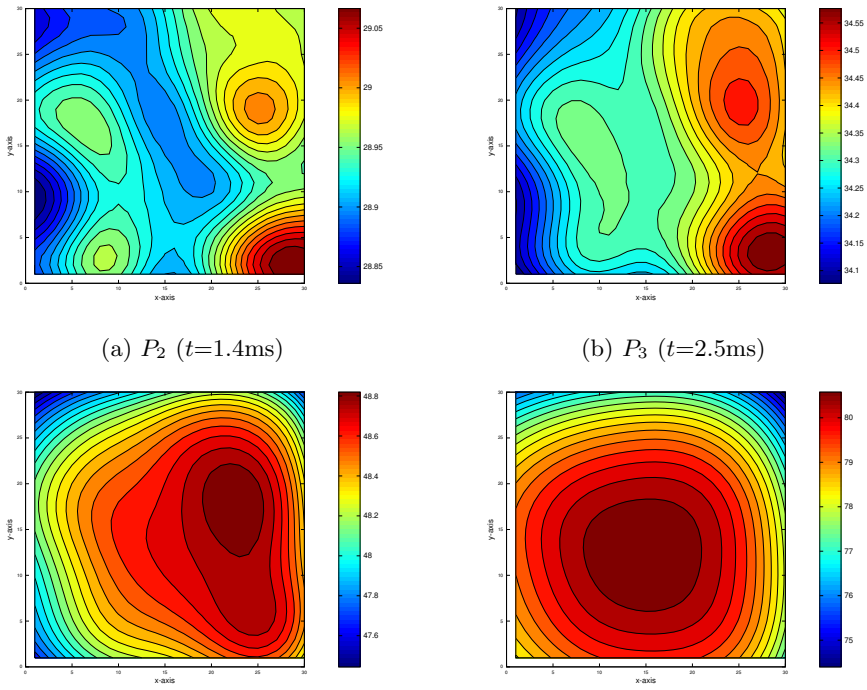
Inputs: $(L, B, I, \kappa, P_1, \cdots, P_M)$
/* $P_1, \cdots, P_M$: thermal profile */
Output: a mapping function ϕ

generate the list of *feasible intervals* for each $P_1, \cdots, P_M$
to produce feasible interval sets $\mathcal{H}_i$, $i = 1, \cdots, M$;
produce $\mathcal{H}_{(1,2)}$ by $\mathcal{H}_1 \cap \mathcal{H}_2$;
for (each $\mathcal{H}_i$, $i = 3, \cdots, M$) {
produce $\mathcal{H}_{(1,2,\cdots,i)}$ by $\mathcal{H}_{(1,2,\cdots,i-1)} \cap \mathcal{H}_i$;
if ($\mathcal{H}_{(1,2,\cdots,i)} = \emptyset$) **return** "no solution";
}
apply *MOSP phase* of WaveMin to $\mathcal{H}_{(1,2,\cdots,M)}$;
return $\phi(\cdot)$ of $H(t_{(\cdot)}) \in \mathcal{H}_{(1,2,\cdots,M)}$ with minimum $p^{max}_{H(\cdot)}$;

Figure 4.2: The procedure of WaveMin-t: considering the effect of thermal variation.

Fig. 4.2 summarizes the procedure of WaveMin-t which consists of three steps: (Step 1) applying the feasible interval generation phase of WaveMin to compute all the feasible intervals of thermal profiles, (Step 2) iteratively intersecting the feasible time intervals to produce a set of feasible time intervals that satisfy the clock skew constraint under all thermal profiles, and (Step 3) applying MOSP phase of WaveMin to find a solution of polarity assignment and buffer sizing with least peak current among the feasible time intervals obtained in Step 2. Since each $\mathcal{H}_j$ contains at most $|L| \cdot (|B| + |I|)$ number of feasible time intervals and the intersection of two feasible time intervals can be computed by $O(|L| \cdot (|B| + |I|))$ with $O(|B| + |I|)$ time for set operation of each $e_i \in L$, The computation time of $\mathcal{H}_{(j,l)}$ is bounded by $O(|L|^3 \cdot (|B| + |I|)^3)$. Thus, the total computation time to $\mathcal{H}_{(1,2,\cdots,M)}$ is bounded by $O(|L|^{(M+1)} \cdot (|B| + |I|)^{(M+1)})$.

4.1.3 Experimental Results



(a) P_2 (t=1.4ms)

(b) P_3 (t=2.5ms)

(c) P_4 ($t = 5.0ms$)

(d) P_5 (t=10ms)

Figure 4.3: Maps of thermal profiles P_2, P_3, P_4, and P_5 for s38584.

To produce a set of thermal map instances, we performed thermal simulation by using the ADI-based thermal simulator package in [39]. For testing ISCAS'89 benchmark circuits, the power density of each thermal node is randomly assigned to a value in between $1.85 \times 10^{14} W/m^3$ and $5.54 \times 10^{14} W/m^3$, as suggested by the example input specification [38] of the simulator. In addition, the position and geometric information is given to the simulator by $\Delta x = 100\mu$m and $\Delta y = 100\mu$m, and the size to contain circuit by 6000μm $\times$ 6000μm. We extract the thermal simulation profiles at the times of 0, 13, 25, 50, 100 and 200 iterations of simulation where we labeled the profiles as P_1, P_2, $\cdots$,

Table 4.1: The temperature information under thermal profiles P_2, P_3, P_4, P_5, and P_6 produced by thermal simulator ADI [38]. The units are in degrees Celsius(°C).

Circuit	Profile P_2		Profile P_3		Profile P_4		Profile P_5		Profile P_6	
	min/max	avg/stdev	min/max	avg/stdev	min/max	avg/stdev	min/max	avg/stdev	min/max	avg/stdev
s13207	27.0/27.3	27.2/0.1	27.6/29.2	28.7/0.4	29.4/35.0	33.2/1.5	33.9/49.4	43.8/4.2	44.0/77.3	64.2/8.9
s15850	27.0/27.3	27.2/0.0	27.5/29.0	28.7/0.3	29.2/34.3	33.3/1.0	33.5/48.1	44.2/3.2	42.9/76.8	66.0/7.6
s35932	27.1/27.3	27.3/0.0	27.8/29.1	28.9/0.2	29.8/34.6	33.9/0.9	35.5/49.2	46.6/2.8	50.7/82.1	74.5/6.9
s38417	27.0/27.3	27.3/0.0	27.7/29.1	28.9/0.3	29.5/34.7	33.8/1.0	34.0/49.5	46.0/3.3	44.4/82.3	71.7/8.5
s38584	27.0/27.3	27.2/0.0	27.6/29.1	28.8/0.3	29.3/34.6	33.6/1.1	33.8/48.9	45.7/3.3	45.0/80.9	71.7/8.4

Table 4.2: The values of peak current and clock skew produced by WaveMin-t with the constraint of skew bound = 50 ps. (The vertical, top to bottom, arrangement of the multiple values at an entry matches the horizontal, left to right, arrangement of the thermal profiles on the corresponding column.)

Circuit	Thermal Profiles									
	(P_1)		(P_1, P_2)		(P_1, P_2, P_3)		(P_1, P_2, P_3, P_4)		$(P_1, P_2, P_3, P_4, P_5)$	
	Peak current (mA)	Clock skew (ps)	Peak current (mA)	Clock skew (ps)	Peak current (mA)	Clock skew (ps)	Peak current (mA)	Clock skew (ps)	Peak current (mA)	Clock skew (ps)
s13207	8.46	49.96	**8.47**	49.32	**8.36**	49.23	**8.98**	49.23	**11.48**	49.23
			8.47	49.29	8.36	49.09	8.98	49.09	11.48	49.09
					8.35	49.18	8.97	49.01	11.46	49.01
							8.96	49.56	11.44	48.98
									11.40	49.90
s15850	0.65	49.85	0.65	49.85	0.65	48.88	0.65	48.88	0.64	48.88
			0.65	49.96	0.65	48.73	0.65	48.63	**0.65**	48.63
					0.65	49.89	0.65	47.68	0.64	47.68
							0.65	49.60	0.64	47.36
									0.62	49.65
s35932	0.02	49.74	0.02	49.74	0.02	48.24	0.02	49.15		
			0.02	49.81	0.02	47.32	0.02	48.46		
					0.02	49.78	0.02	48.21		
							0.02	49.93		
s38417	0.11	49.98	0.11	49.54	0.11	48.75	**0.11**	49.77		
			0.11	49.86	0.11	48.02	0.11	48.72		
					0.11	49.79	0.11	45.66		
							0.11	49.20		
s38584	0.12	49.89	0.12	49.89	0.12	49.87	0.11	49.68		
			0.12	49.65	0.12	48.74	0.11	48.24		
					0.12	50.00	0.11	46.46		
							0.12	48.91		

P_6, respectively. We set the time increment parameter Δt in the simulator [39] to 100ns, thus the duration of circuit execution for the last profile P_6 being $t =$ 20ms. For example, Fig. 4.3 shows the thermal maps of P_2, P_3, P_4 and P_5 for circuit s38584. The minimum and maximum temperature, and the average and standard deviation of the temperature under each thermal profile are shown in Table 4.1. With the assumption that the thermal variance has negligible effect on unit length capacitance, we calculate the interconnection wire resistance per unit length by equation [35]:

$$r = \rho_0\{1 + \beta \cdot T(x, y)\} \tag{4.2}$$

where ρ_0 is the unit resistance per unit at 0^oC, β is the temperature coefficient of resistance $(1/^o\text{C})$, and $T(x, y)$ is the temperature at point (x, y). In this experiment $\beta = 0.0068(1/^o\text{C})$ [40]. For wire model, the π network is used for simulation as TACO algorithm [36] does.

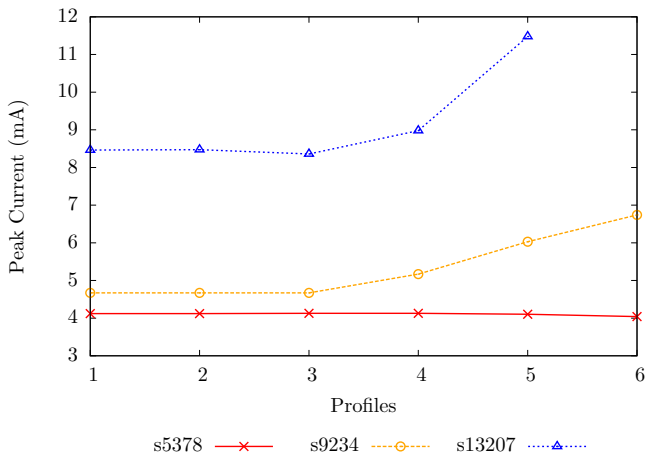


Figure 4.4: The curves showing the changes of peak current values as the number of thermal profiles considered increases from P_1 only (marked as P_1), P_1 and P_2 only (marked as P_2), $\cdots$, finally P_1 through P_6 (marked as P_6 with skew bound = 50 ps (i.e., results in Table 4.2).

WaveMin-t is then applied to each of thermal profiles, followed by performing SPICE simulation to produce the noise data. Table 4.2 shows the values of peak currents and clock skews for different sets of profiles under the clock skew bound of 50 ps. The red colored number in each entry of peak current column indicates the worst peak value among the profiles in the corresponding column. From the two tables, we observe a consistent trend: the peak current increases (or decreases) as more (or less) thermal profiles are considered. Finally, Fig. 4.4 shows how the peak current values change as the circuit execution is performed, starting from considering P_1 only, considering P_1 and P_2 only, $\cdots$, finally considering P_1 through $P6$ for circuits s5378, s9234, and s13207 with skew bound = 50 ps.

4.2 Coping with Delay Variations

4.2.1 Introduction

In sub-45nm CMOS technology nodes, the effect of process variation is one of the most important factors that must be taken into account in clock tree synthesis/optimization. The clock signal arrival times at the sinks and the clock skews are random variables and their deviations are becoming more and more difficult to control as the technology scales down. Process variations of channel length/width, oxide thickness, threshold voltage, and wire width/thickness affect the delay variation of interconnect produces up to 25% of clock skew variation [41].

Traditionally, worst-case timing analysis is used to consider the delay variation caused by the process variation. However, as the delay variation increases, the timing margin given by designer based on the analysis occupies a significant portion of clock timing, causing to degrade circuit performance. To cope with the worst-case timing analysis, the statistical static timing analysis (SSTA)

has been developed. By computing the delays as random variables, the excess margins were successfully removed [42, 43, 44].

In this section, we demonstrate that the proposed polarity assignment framework can be extended for yield aware polarity assignment. Although several methods of polarity assignment had been proposed, relatively less attention has been paid to process variations. In [25], Lu and Taskin reported clock skew at the worst corner. By greedily finding the paths that have the greatest difference of clock arrival times and tuning the buffer polarity associated with the paths after the initial clock polarity assignment, they were able to trade-off the worst corner clock skew with increased noise. In [13], Kang and Kim proposed a more systematic approach. They used statistical static timing analysis (SSTA) on the clock tree to examine the yield of *each* pair of the leaf buffering elements. Precisely, they calculated the statistic arrival time difference for each pair of leaf buffers which were optimized to satisfy the yield constraint, while noise was minimized heuristically. However, the heuristic has no direct control over the *design yield*, which is the global clock skew of the whole clock tree.

4.2.2 The Impact of Process Variations on Polarity Assignment

Table 4.3: The impact of process variations on clock skew

Circuit	μ (ps)		σ (ps)	
	PeakMin	WaveMin	PeakMin	WaveMin
s13207	5.88	5.88	1.58	1.59
s15850	42.99	4.53	10.94	1.97
s35932	106.2	120.69	11.58	14.02
s38417	94.72	119.61	11.76	11.97
s38584	80.45	85.5	10.17	9.86
ispd09f31	78.68	78.31	15.36	15.15
ispd09f34	68.48	65.86	15.11	15.14

Monte Carlo simulations were run on the clock trees obtained by PeakMin

Table 4.4: The impact of process variations on peak current noise

Circuit	μ (mA)		σ (mA)	
	PeakMin	WaveMin	PeakMin	WaveMin
s13207	7.56	7.56	0.36	0.36
s15850	1.15	3.11	0.1	0.23
s35932	24.2	14.82	0.99	1.73
s38417	16.65	12.54	0.38	0.44
s38584	16.86	16.41	0.52	0.71
ispd09f31	62.04	65.07	4.34	2.77
ispd09f34	42.8	44.11	3.22	3.28

[14] and WaveMin in Chapter 2, where the trees were optimized with $\kappa = 100$ ps and $|S| = 158$, for the investigation of the process variations on the optimized clock trees. Wire widths, wire lengths, buffer/inverter widths, and threshold voltages were randomized in which all the variables follow the Gaussian random distribution of $N(\mu, \sigma^2)$, where μ is the variables' respective nominal value and σ satisfies $\sigma/\mu = 5\%$. For each benchmark circuit, 1000 randomized instances were generated for HSPICE simulations.

On average, 95.5% and 83.9% of the clock trees produced by PeakMin and WaveMin satisfied the clock skew bound κ, respectively. This can be observed in Table 4.3; the results by WaveMin has larger average clock skews. This is because some of the circuits optimized by WaveMin had the nominal clock skews that were very close to κ, so that they were more sensitive to the variations; WaveMin tries to disperse the noise waveform over time slots, but this leaves less room for variations.

4.2.3 Proposed Method for Variation Resiliency

Here, we propose a design yield aware polarity assignment heuristic, UsefulMin-V. The design yield is defined as the probability of the whole clock tree satisfying the global clock skew constraint κ. Given the design yield constraint γ, we make

the following modifications to UsefulMin:

- In mapping the problem to a maximum clique problem, we create an edge when the pair of vertices in the graph satisfy the clock tree yield constraint γ.
- During the local search, the cliques now have two parameters *noise* and *yield*.
  - When the current best clique doesn't satisfy the design yield γ, the clique with higher yield is kept as the best clique.
  - When the current best and the neighbor cliques both satisfy γ, we keep the one with lower *noise*.

The γ parameter constraining each edge in the first step provides initial filtering, since even one leaf pair that does not satisfy γ is enough to lower the yield of the whole clock tree below γ. This corresponds to finding pair choices that meet the pairwise yield constraint in Kang's [13] algorithm.

The design yield is verified in the second step for each clique found in the graph. This ensures that yield γ is satisfied for the whole clock tree. The final yield depends on the initial clique at the start of the local search. To improve the resulting yield, the local search may be started from the unoptimized clock tree, as the single polarity clock tree is likely to have a high yield.

4.2.4 Experimental Results

Since both skew tuning [25] and pairwise [13] methods are incapable of buffer sizing, we defined $B = \{BUF\_X8\}$ and $I = \{INV\_X4\}$. INV_X4 was chosen so, as it had the closest matching clock signal propagation delay to BUF_X8. Like other experiments, useful clock skew constraints were randomly generated

Table 4.5: Average peak current of the optimized clock trees by skew tuning [25], pairwise optimization [13] and UsefulMin-V.

Benchmark Circuit	γ	Average Peak Current (mA)		
		Tuning [25]	Pairwise [13]	UsefulMin-V
01	0.83	120.10	123.68	92.93
02	0.39	222.75	230.50	195.54
03	0.98	50.40	51.10	51.47
04	0.98	54.79	55.43	58.77
05	0.98	27.93	27.78	27.92
06	0.98	39.54	39.65	40.12
07	0.98	59.76	62.09	66.45
08	0.98	47.59	47.45	45.40

Table 4.6: Design yield of the optimized clock trees by skew tuning [25], pairwise optimization [13] and UsefulMin-V.

Benchmark Circuit	γ	Design Yield (%)		
		Tuning [25]	Pairwise [13]	UsefulMin-V
01	0.83	76.4	73.1	81.1
02	0.39	28.6	27.2	39.4
03	0.98	94.9	93.8	98.6
04	0.98	94	94.4	98.9
05	0.98	98.8	98.6	99.7
06	0.98	99.4	99.7	100
07	0.98	96.3	96.5	99.1
08	0.98	99.7	99.8	99.8

in the range of [60, 90] ps. In the experiments, we assumed that buffer/inverter and interconnect delays are spatially correlated normally distributed random variables. Spatial correlations were modelled using the grid model proposed in [42]. Each 3σ value of the distributions were set to 5% of their nominal delays. During exploration, design yield was computed using statistical *max* operation as proposed in [42]. Given (correlated) normal distributions d_1, d_2, d_3, ..., $max(d_1, d_2, d_3, ...)$ operation computes approximated normal distribution of the maximum value.

Tables 4.5 and 4.6 summarizes the results of variation aware clock polarity assignment. Design yield is obtained by running Monte Carlo simulation on 1000 randomized instances of the clock tree. γ column shows the yield constraint input to the algorithms. In some cases, UsefulMin-V algorithm fails to meet γ constraint by a few percent point. This is attributed to the fact that the statistical max operation is an approximation rather than the true distribution. However, it is evident that UsefulMin-V is more capable of keeping the yield constraint compared to the other two algorithms. In all circuits, UsefulMin-V maintains comparable noise to other methods while keeping γ. Particularly in circuits 01 and 02 of ISPD'10, UsefulMin-V reduced considerable noise while maintaining higher yield than other algorithms. This shows that the useful skew approach proposed in this work can exploit the individual skew constraints to reduce noise, even under clock delay variations.

4.3 Coping With Multi-Mode Designs

4.3.1 Introduction

The conflicting high-performance and low-power requirements imposed to the designers lead to the introduction of advanced low power techniques, such as Dynamic Voltage Frequency Scaling (DVFS), on real designs. Such design tech-

niques require the chip to operate in multiple power modes, where in each mode, the subareas partitioned by the voltage islands can operate at different voltages. As the clock tree spans across multiple voltage islands, the different supply voltages in the voltage islands can cause clock skew violations unless the clock tree is carefully designed. In this section, we use the concept of *intersection of intervals* proposed in Section 4.1 to satisfy the clock skew constraint and provide a method to minimize the clock noise in multi-power mode designs.

4.3.2 Proposed Method

Table 4.7: Characterization of B = {BUF_X1, BUF_X2} and I = {INV_X1, INV_X2}. T_D represents the signal propagation delay, P+ and P- indicate the values of the peak I_{DD} at the rising and falling edges of the input. (For brevity, we omit here the values of P+ and P- of I_{SS}.)

Type	$V_{\mathrm{DD}} = 0.9$ V			$V_{\mathrm{DD}} = 1.1$ V		
	T_D	P+	P-	T_D	P+	P-
BUF_X1	27	120	10	24	130	13
BUF_X2	23	234	36	19	255	44
INV_X1	24	10	120	21	13	130
INV_X2	22	36	234	17	44	255

Consider the example of clock tree shown in Fig. 4.5 with two power modes M_1 and M_2 such that in M_1, both of the voltage islands $A1$ and $A2$ operate at $V_{\mathrm{DD}} = 1.1$ V, by which all leaf nodes (i.e., sinks) have arrival time of 70, while in M_2, $A2$ operates at $V_{\mathrm{DD}} = 0.9$ V, which increases the arrival times of e_3 and e_4 from 70 to 78 (+4 from the parent node of e_3 and e_4 and another +4 from each of e_3 and e_4). The clock tree must support both M_1 and M_2 under some bounded clock skew constraint. Let the skew bound κ be 5 in this example. Clearly, the clock skew in Fig. 4.5 is violated in M_2.

To tackle this problem, we first compute the sets of feasible intervals for all power modes, and then *intersect* them to identify, for each sink in L, the

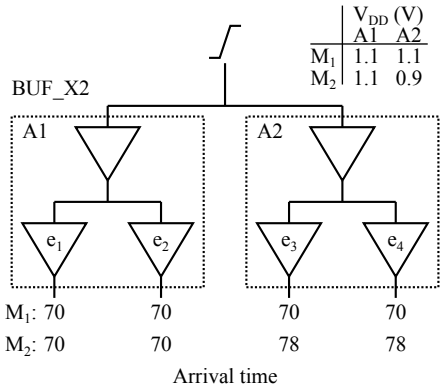


Figure 4.5: An example of clock tree which has two voltage islands $A1$ and $A2$ such that in power mode M_1, both $A1$ and $A2$ operate at $V_{\mathrm{DD}} = 1.1$ V and in power mode M_2, $A1$ operates at 1.1 V while $A2$ operates at 0.9 V. All nodes are initially assigned with BUF_X2.

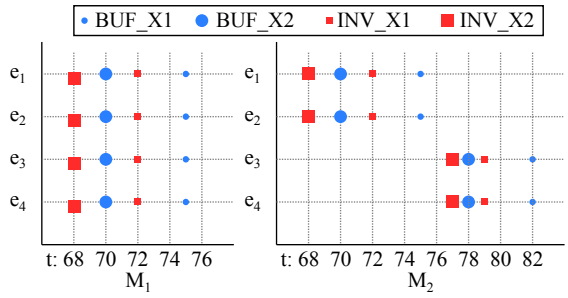


Figure 4.6: Illustration of intervals of arrival times for the example in Fig. 4.5 and Table 4.7. Each dot in the grids represents a buffer or inverter. For example, the large red dot located at position (68, e_3) in M_1 indicates that e_3 has arrival time of 68 when INV_X2 is assigned to it in power mode M_1.

buffer/inverter types in $B \cup I$ that can be assigned to the sink. For example, Fig. 4.6 illustrates all intervals for power modes M_1 and M_2 in Fig. 4.5. With $\kappa = 5$, in M_1 there are time intervals [70, 75], [67, 72], [65, 70], and [63, 68] defined by arrival times 75, 72, 70, and 68, and all of them are feasible intervals. In M_2, there are 8 intervals but only intervals [74, 79], [73, 78], and [72, 77] are feasible. With feasible intervals in all power modes, we are now ready to obtain intersections of feasible intervals in different power modes. Fig. 4.6 involves 12 intersections between M_1 and M_2 i.e., {[70, 75], [67, 72], [65, 70], [63, 68]}×{[74, 79], [73, 78], [72, 77]}. For example, intersection (70, 79) (= [65, 70]×[74, 79]) denotes that interval [65, 70] of M_1 and [74, 79] of M_2 are chosen, which means to extract, for each sink, a maximal subset of buffers and inverters that are contained in both of the sets of feasible buffers and inverters in [65, 70] of M_1 and [74, 79] of M_2. In Fig. 4.6, since [65, 70] of M_1 has {BUF_X2, INV_X2} for sink e_1, {BUF_X2, INV_X2} for e_2, {BUF_X2, INV_X2} for e_3, and {BUF_X2, INV_X2} for e_4 while [74, 79] of M_2 has {BUF_X1} for e_1, {BUF_X1} for e_2, {BUF_X2, INV_X1, INV_X2} for e_3, and {BUF_X2, INV_X1, INV_X2} for e_3, intersection (70, 79) returns ϕ (= {BUF_X2, INV_X2}∩{BUF_X1}) for e_1, ϕ (= {BUF_X2, INV_X2}∩{BUF_X1}) for e_2, {BUF_X2, INV_X2} (= {BUF_X2, INV_X2}∩{BUF_X2, INV_X1, INV_X2}) for e_3, and {BUF_X2, INV_X2} (= {BUF_X2, INV_X2}∩{BUF_X2, INV_X1, INV_X2}) for e_4. An intersection $(t_i, \cdots, t_j)$ is called a *feasible intersection* if the resulting set of buffers and inverters for every sink is not empty and called an *infeasible intersection*, otherwise. Thus, (70, 79) is an infeasible intersection.

The example in Fig. 4.6 has three feasible intersections (75, 79), (75, 78) and (72, 77) among 12 possible intersections. The intersection results are summarized in Table 4.8 where *fsbl* indicates that its buffer or inverter is feasible to use in that interval and *infsbl* indicates that it is not feasible. As long as

Table 4.8: Node-to-type feasibility information of all feasible intersections, when the clock skew bound is $\kappa = 5$.

Intersection	Node	BUF_X1	BUF_X2	INV_X1	INV_X2
(75, 79)	e_1	**fsbl**	infsbl	infsbl	infsbl
	e_2	**fsbl**	infsbl	infsbl	infsbl
	e_3	infsbl	**fsbl**	**fsbl**	infsbl
	e_4	infsbl	**fsbl**	**fsbl**	infsbl
(75, 78)	e_1	**fsbl**	infsbl	infsbl	infsbl
	e_2	**fsbl**	infsbl	infsbl	infsbl
	e_3	infsbl	**fsbl**	infsbl	infsbl
	e_4	infsbl	**fsbl**	infsbl	infsbl
(72, 77)	e_1	infsbl	infsbl	**fsbl**	infsbl
	e_2	infsbl	infsbl	**fsbl**	infsbl
	e_3	infsbl	infsbl	infsbl	**fsbl**
	e_4	infsbl	infsbl	infsbl	**fsbl**

fsbl: assignment with no skew violation
infsbl: assignment that causes skew violation

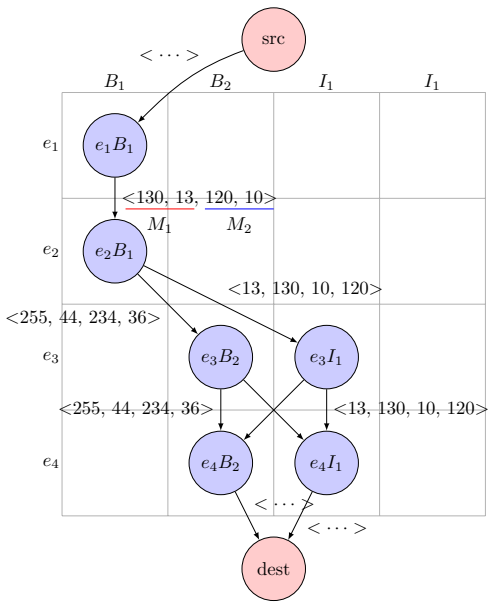


Figure 4.7: The updated MOSP graph supporting intersection (75, 79) in Fig. 4.6. The cost formulation of MOSP problem is still vaild.

only the feasible types are selected, the clock skew is satisfied for all power modes. The difficulty lies in minimizing the noise for multiple modes as there are multiple different noise values from multiple modes to optimize. In this noise optimization problem, the objective is to minimize the worst case noise. In other words, noises in M_1 and M_2 for the example in Figs. 4.5 and 4.6 have the same priority or weight; if we concatenate the noise values from all the modes into one vector, this is still a valid cost formulation of MOSP problem. Hence, we translate the noise from each power mode as an extra dimension in the MOSP problem formulation. Fig. 4.7 shows the MOSP graph of the intersection (75, 79). As with optimization of single power mode, MOSP graph vertices represent which buffer or inverter types are available to each sink. The arc weights are composed of noise from multiple modes. For example, the arc from e_1B_1 to e_2B_1 has weight of <130, 13, 120, 10> where 130 and 13 are from P+ and P- columns of $V_{\mathrm{DD}} = 1.1$ V and 120 and 10 are from $V_{\mathrm{DD}} = 0.9$ V in BUF_X2 row of Table 4.7. Optimizing this MOSP problem (without approximation) yields noise of <268, 268, 280, 266> with the assignment of BUF_X1 to e_1, BUF_X1 to e_2, INV_X1 to e_3, and INV_X1 to e_4, resulting in clock skew of 3 in M_1 and 4 in M_2. Thus, the worst noise for the feasible intersection (75, 79) is 280. Likewise, the worst noises for the other intersections (75, 78) and (72, 77) are each 770. Consequently, the best solution is from (75, 79) since its noise is the least.

Although WaveMin can endure some degree of clock skew, the arrival time variation may be too large in designs of multiple power modes, so that it is impossible to satisfy the clock skew without the use of Adjustable Delay Buffers (ADBs). ADBs are buffers whose signal propagation delays can be adjusted at runtime. A capacitor bank based implementation of ADB is illustrated in Fig. 4.9. The capacitor bank controller controls the number of active capacitors

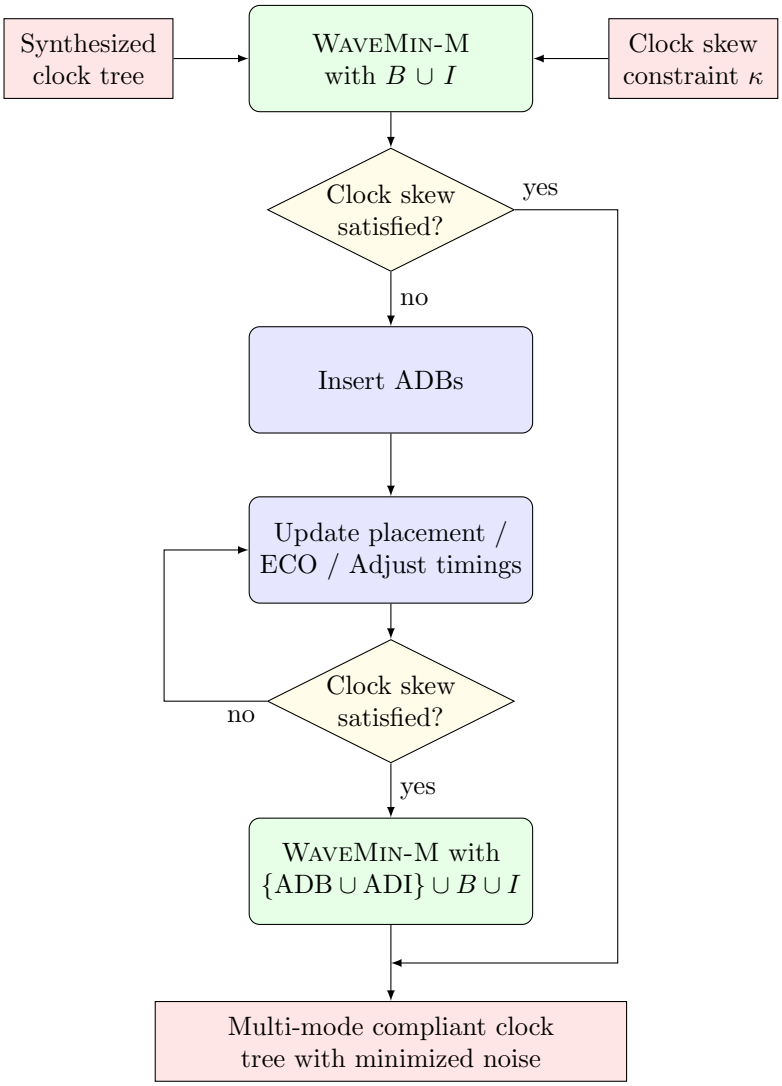


Figure 4.8: The flow of WaveMin-M, an extension of WaveMin to support multiple power mode designs. Note that module Insert ADBs resolves the clock skew violation, and the subsequent module WaveMin performs the polarity assignment with library $B \cup I \cup ADB \cup ADI$ to reduce the noise while retaining the satisfaction of clock skew constraint. Also, it is the responsibility of the ADB insertion algorithm/method to update placement (the embedding of the ADBs is likely to cause placement changes) and meet timing closure.

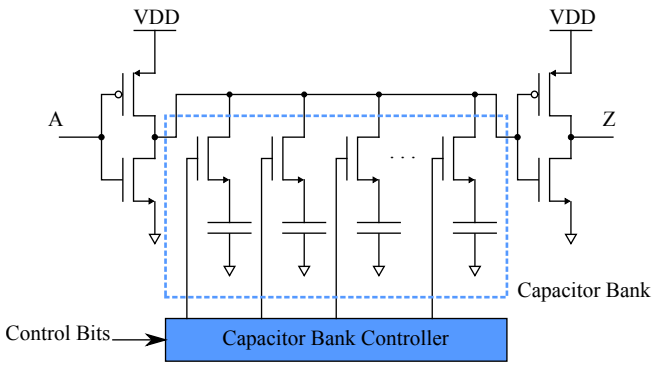


Figure 4.9: The schematic of a capacitor bank based adjustable delay buffer (ADB).

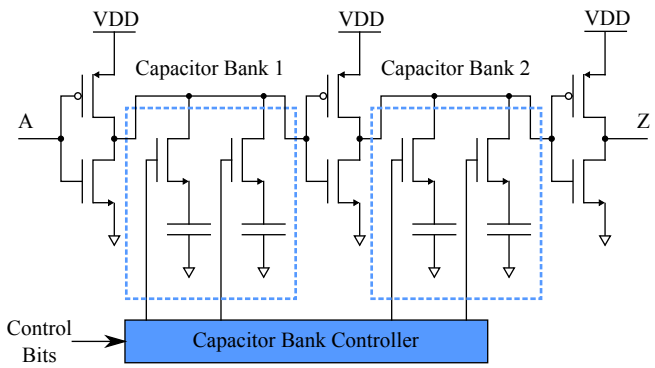


Figure 4.10: The proposed capacitor bank based implementation of adjustable delay inverter (ADI). The capacitor banks contain switched capacitors which are dynamically controllable. The number of capacitors in the two banks is a design parameter that controls the granularity of the discrete delay steps and the delay range of the ADI.

between the inverters. As the number of active capacitors increases, the propagation delay of the ADB increases. Fig. 4.8 is the flow of WaveMin-M, an extension of WaveMin for multiple power mode designs. Given a synthesized clock tree and clock skew constraint κ, the clock signal arrival times in each power mode is calculated by WaveMin and noise is minimized, if it is possible to satisfy κ with only polarity adjustments and buffer/inverter sizing. If it fails, ADBs are inserted to satisfy κ, then WaveMin is executed again, in which the inverter library I contains an ADI (adjustable delay inverter) in Fig. 4.10 as well as the normal inverters of different size. Note that ADBs that have been already allocated must not be replaced with buffers or inverters since ADBs are essential to meet the clock skew bound in multiple power modes; each ADB can be replaced with an ADI or stay as ADB. Likewise, non-ADBs may not become ADBs or ADIs since this replacement leads to unnecessary increase of area. This restriction is handled during feasible buffer/inverter type computation by checking if the leaf node is an ADB or not. After the ADB insertion, at least one WaveMin solution exists for the ADB inserted clock tree – the trivial solution in which no buffer sizing and polarity assignment are applied.

One of the bottlenecks of this optimization is the intersection process. In [14], the time complexity of the intersection process is $O(|L|^{(M+1)} \cdot (|B| + |I|)^{(M+1)})$ where M is the number of power modes. The complexity increases exponentially as the number of modes increases. In thermal mode, this was a less concern since only a few coolest and hottest modes may be considered. Although even the brute force method may have a fast execution time in practice, depending on the input size, – this is because most of the intersections is not feasible and pruned early during execution – it is possible to improve the performance through the use of the concept of *degree of freedom*: given a feasible intersection, the degree of freedom is calculated by simply counting the

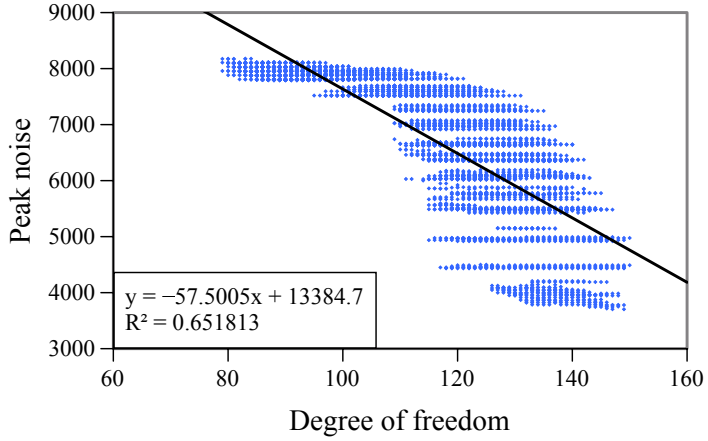


Figure 4.11: The relationship between peak noise and the degree of freedom which measures the flexibility of polarity assignment of a feasible intersection. The plot has been acquired by optimizing s35932 circuit in ISCAS'89 benchmark set.

total number of the buffers and inverters produced by the intersection for all sinks. For instance, in Table 4.8, the degree of freedom of intersection (75, 79) is 6 and (75, 78) is 4. As illustrated in Fig. 4.11, it is observed that there is a negative correlation between the degree of freedom and peak noise: the more the freedom is, the lower the noise is. Hence, we use the degree of freedom to prune out less free intersections during the intersection process.

4.3.3 Experimental Results

WaveMin-M was applied to the benchmark circuits, given four power modes. Each benchmark was partitioned into 4 to 10 power domains with each having two operating modes at supply voltage levels of 0.9 V and 1.1 V. Table 4.9 summarizes the results of WaveMin-M. While any ADB embedding algorithms may be used, we employed the algorithm in [45], which is known to insert a minimum number of ADBs in multiple power modes to resolve the clock

Table 4.9: The result produced by WaveMin-M that supports designs with multiple power modes.

Bench-mark ckt.	Skew Bound (ps)	ADB-embedding-only [14]				WaveMin-M					Improvement		
		Peak curr. (mA)	V_{DD} noise (mV)	Gnd noise (mV)	#ADBs	Peak curr. (mA)	V_{DD} noise (mV)	Gnd noise (mV)	#ADBs	#ADIs	Peak curr. (%)	V_{DD} noise (%)	Gnd noise (%)
s13207	90	21.53	6	6.53	33	18.91	5.68	8.11	33	0	12.17	5.33	-24.20
	110	20.12	5.82	7.95	33	17.37	5.83	6.7	33	0	13.67	-0.17	15.72
	130	19.71	5.42	7.21	53	19.12	6.99	8.11	53	0	2.99	-28.97	-12.48
s15850	90	8.77	2.4	3.78	18	7.33	2.92	2.81	18	0	16.42	-21.67	25.66
	110	8.83	2.27	3.67	33	8.33	2.65	3.14	33	0	5.66	-16.74	14.44
	130	8.88	2.26	3.85	0	8.36	3.46	3.26	0	0	5.86	-53.10	15.32
s35932	90	99.27	29.84	26.7	75	104	33.03	27.96	69	6	-4.76	-10.69	-4.72
	110	100.4	30.98	26.66	164	93.05	29.93	24.71	164	0	7.32	3.39	7.31
	130	97.44	30	26.38	293	78.39	23.93	23.24	293	0	19.55	20.23	11.90
s38417	90	74.94	22.05	20.43	113	69.46	21.62	21.5	109	4	7.31	1.95	-5.24
	110	80.4	23.88	21.61	127	84.55	27.1	19.51	127	0	-5.16	-13.48	9.72
	130	78.4	23.72	17.51	288	53.46	18.08	22.93	288	0	31.81	23.78	-30.95
s38584	90	56.06	16.12	15.72	279	51.45	15.75	15.81	213	66	8.22	2.30	-0.57
	110	58.23	15.36	16.16	167	51.01	14.95	16.25	101	66	12.40	2.67	-0.56
	130	57.91	16.75	16.57	275	56.02	16.86	15.02	261	14	3.26	-0.66	9.35
f31	90	90.71	0.54	0.94	12	62.65	0.27	0.51	12	0	30.93	50.00	45.74
	110	92.29	0.54	0.94	2	51.36	0.5	0.9	2	0	44.35	7.41	4.26
	130	79.27	0.53	0.94	69	47.97	0.28	0.51	69	0	39.49	47.17	45.74
f34	90	65.17	0.53	0.93	18	49.88	0.52	0.92	18	0	23.46	1.89	1.08
	110	50.03	0.53	0.94	32	36.28	0.51	0.89	32	0	27.48	3.77	5.32
	130	57.32	0.53	0.92	31	33.54	0.27	0.5	31	0	41.49	49.06	45.65
Average											**16.38**	3.50	8.50

skew violations[3]. The optimization results produced by WaveMin-M have been compared with the noise-unaware clock trees (denoted as *ADB-embedded-only* in Table 4.9) produced by [45] which inserts ADBs to meet the clock skew constraint for every power mode. It is evident from the table that WaveMin-M reduces noise on multiple power mode designs, without violating clock skew bound. On average, WaveMin-M achieves 16.38% peak current reduction. One interesting data to note is s15850 with skew bound of 130 ps. It has no ADB allocated, yet the buffer sizing managed to satisfy the clock skew constraint for all modes.

As a side effect of embedding ADBs, loosening the clock skew bound does not always reduce the clock noise, as it increases from 90 ps to 130 ps. This is due to the ADB embedding algorithm [45] sharing the clock skew bound constraint with WaveMin-M. The ADB embedding is done first, which exploits clock skew bound for the reduction of the number of embedded ADBS, leaving less room for WaveMin-M, which takes advantage of the clock skew to reduce noise.

The reasons that only a fraction of ADBs were replaced with ADIs is that (1) while ADBs are located at both leaf and non-leaf positions, only the ones at the leaf positions are subject to WaveMin and may be replaced with ADIs; (2) since ADIs have longer signal propagation delay than that of ADBs, during feasible type computation, ADIs were mostly pruned. As shown in Fig. 4.10, there are three inverters in an ADI which causes ADIs to have longer delays than ADBs. Currently in our implementation, the first inverter which directly receives the incoming clock signal has NMOS width of 45nm which is the smallest feature size allowed by the technology. Thus, it is impossible to reduce the ADI size. Instead, the designer might choose to have larger ADBs so that the signal

[3] Since ADBs are large, insertion of ADBs is likely to cause placement changes which will require placement update / ECO / timing adjustments. However, in this experiment, we only simulated the clock tree and justification steps were omitted.

propagation delay is balanced. However, this will cause ADBs to occupy larger area. Thus, in this experiment, we chose to have the unbalanced ADBs and ADIs.

4.4 Orthogonality with Other Design Techniques – Clock Gating

4.4.1 Introduction

One of the powerful features of clock polarity assignment technique is that it is orthogonal with many other clock optimization techniques, as it only affects the polarity (and the timings) of the clock signals. In this section, we demonstrate the orthogonality by applying the polarity assignment technique in conjunction with clock gating technique.

Several techniques to reduce the dynamic power are developed and clock gating is one of the most common techniques applied to IC products, as the technique is readily available in Electronic Design Automation tools. When a flip-flop (FF) is clocked, it consumes dynamic power regardless of the input data, even when the input data does not switch. With clock gating, the clock signals are ANDed with predefined enabling signals, *gating* the clock signal. This saves the dynamic power consumed at the FFs. Clock gating technique is available at many levels of the design: system architecture, block design, logic design, and gate levels [46, 47].

4.4.2 Proposed Partitioning Method

Since the two techniques are orthogonal, they can be applied independently to the clock trees. However, they are not completely transparent to one another: the clock gating technique affect the result of the clock polarity assignment by partitioning the clock tree into several gated subtrees. This gives three cases of

leaf buffer partitioning method for WaveMin or UsefulMin.

1. Clock gate cluster unaware polarity assignment, using only zones, as it was done in the previous sections.

2. Cluster the leaf buffering elements by gate cluster, ignore zones.

3. Consider both zones and the gate clusters. That is, partition the leaves by zones and for each zone, partition the zones into subzones by gate clusters.

In design flow, we applied the clock gating technique first, so that the gate clusters are initiated. Next, we applied the WaveMin, using one of the three leaf buffer partitioning method presented above.

4.4.3 Experimental Results

The proposed algorithm WaveMin was implemented in C++ language on a Linux machine. Clock trees were generated for ISPD'10 high performance clock network synthesis contest benchmarks with the algorithm in [20], using Nangate 45nm Open Cell Library [19]. Since ISPD'10 benchmarks have only clock sink information and no circuit/individual clock skew constraint information, clock gating was randomly generated with $\log_2 N$ clock gates, where N is the number of clock buffering elements in the clock tree. 8 clock gating modes were generated, where 7 were generated randomly and one is the high performance mode with all clock gates letting the clock signal through. All of the 8 modes were unique that none of them had the same clock gate configuration.

Table 4.10 shows the experimental results of ISPD'10 benchmark circuits 04, 05. Even though the experiments were done in all circuits, only the two circuits are presented here, as the rest of the circuits reveal the same trend

Table 4.10: Results of ISPD'10 benchmark circuits 04, 05.

ISPD'10 Benchmark Circuit	Gating mode	Peak current noise (A)		
		Locality	Gated clusters	Locality + Gated clusters
04	No gating	0.059	0.050	0.050
	1	0.059	0.050	0.050
	2	0.055	0.046	0.047
	3	0.056	0.048	0.049
	4	0.056	0.047	0.048
	5	0.059	0.049	0.050
	6	0.057	0.048	0.050
	7	0.056	0.048	0.049
05	No gating	0.037	0.027	0.031
	1	0.033	0.024	0.028
	2	0.020	0.015	0.018
	3	0.033	0.024	0.027
	4	0.035	0.025	0.030
	5	0.023	0.017	0.021
	6	0.038	0.028	0.031
	7	0.035	0.026	0.029

as 04, 05. As expected, when all of the clock gates are letting the clock signal through (no-gating mode), all of the buffering elements are activated, making this mode the worst case scenario mode. This observation holds for all three of the leaf buffering element partitioning method in all benchmark circuits.

Table 4.11: The effectiveness of leaf buffering element partitioning method.

ISPD'10 Benchmark Circuit	# Gates	Peak current noise (A)		
		Locality	Gated clusters	Locality + Gated clusters
01	11	0.005	0.005	0.005
02	12	0.003	0.003	0.003
03	11	**0.046**	**0.046**	0.048
04	11	0.059	**0.050**	**0.050**
05	11	0.037	**0.027**	0.031
06	10	0.040	**0.038**	**0.038**
07	11	0.065	0.068	**0.064**
08	11	0.043	**0.042**	0.043

Given the fact that the worst case scenario is the ungated mode, condensing the noise of ungated modes into one table yields Table 4.11. There are only trivial differences in the noise values between the three partitioning methods. This demonstrates that the two techniques, clock polarity assignment and clock gating, are orthogonal so that the designer may apply clock polarity assignment without the information from clock gating.

4.5 Summary

In this chapter, various extensions of clock polarity assignment were presented. The flexibility of the WaveMin and UsefulMin problem formulation is shown to successfully embrace the challenging operating conditions, such as thermal variations, process variations and multi-corner multi-mode design regime. In addition, the orthogonality of the polarity assignment technique, which makes

the technique easy to apply in practice, is demonstrated by applying both clock gating and polarity assignment design technique onto one clock tree.

Chapter 5

Conclusion

The contributions of this dissertation is summarized as follows.

5.1 Clock Polarity Assignment Under Bounded Skew

In the chapter, a comprehensive graph-based algorithm for solving clock polarity assignment problem combined with buffer/inverter sizing, that supports fine-grained peak current noise model was proposed. The experimental results show that the algorithm reduced the peak noise by 15.62% on average, over that by the best known method with coarse-grained noise model [14]. This is due to the fact that the fine-grained model allows better exploitation of the clock skew to further reduce the clock noise.

In addition, while there were many methodologies developed for assigning clock polarity, no attention had been the voltage fluctuations on the power delivery network. To our knowledge, this is the first work in clock polarity assignment to report frequency domain properties of the voltage noise. These results would inspire the development of polarity algorithms that can better

consider the voltage fluctuations in the power delivery network.

5.2 Clock Polarity Assignment Under Useful Skew

In the chapter, a scalable solution to the problem of the clock polarity assignment under useful clock skew constraints is proposed. Unlike the conventional (global) clock skew bound constrained approaches, the new method exploited individual clock skew constraints to further reduce the peak current. Precisely, we formulated the problem into the maximal clique exploration problem and employed a K-neighbor search scheme to trade-off the run time and quality of polarity assignment. For designing high speed systems with tight time margin, the proposed approach would be useful in mitigating the clock noise, which otherwise the conventional polarity assignment approaches could rarely achieve.

5.3 Extensions of Clock Polarity Assignment

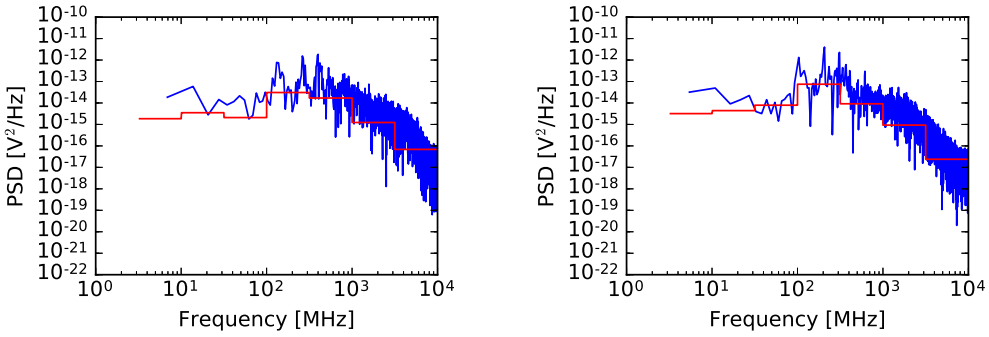
In this chapter, various extensions of clock polarity assignment were presented. The flexibility of the WaveMin and UsefulMin problem formulation is shown to successfully embrace the challenging operating conditions, such as thermal variations, process variations and multi-corner multi-mode design regime. In addition, the orthogonality of the polarity assignment technique, which makes the technique easy to apply in practice, is demonstrated by applying both clock gating and polarity assignment design technique onto one clock tree.

Appendices

Appendix A

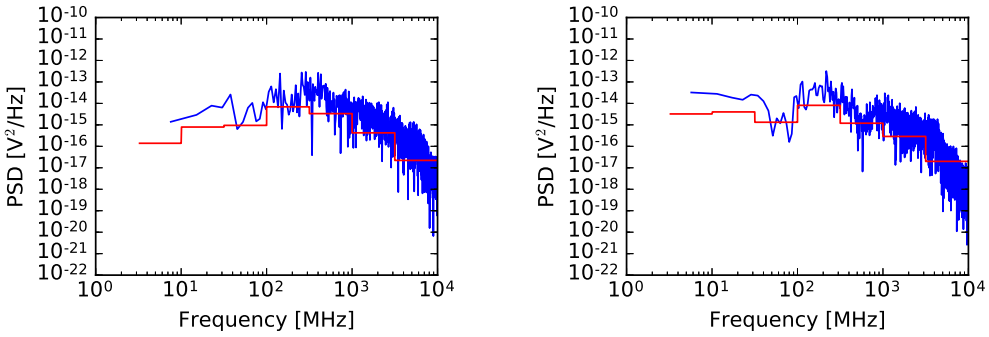
Power Spectral Densities of ISCAS'89 Circuits

The red bars show the average power in the given frequency band. Both the unoptimized and optimized circuits have the peak powers around 100-1000 MHz band. However, WaveMin tends to reduce power noise at higher frequencies ($>$ 100 MHz). Although the average noise power at lower frequencies have increased, considering that the horizontal axis is in log scale, the overall noise power have decreased. Decoupling capacitor of 30 fF was used to acquire these plots.



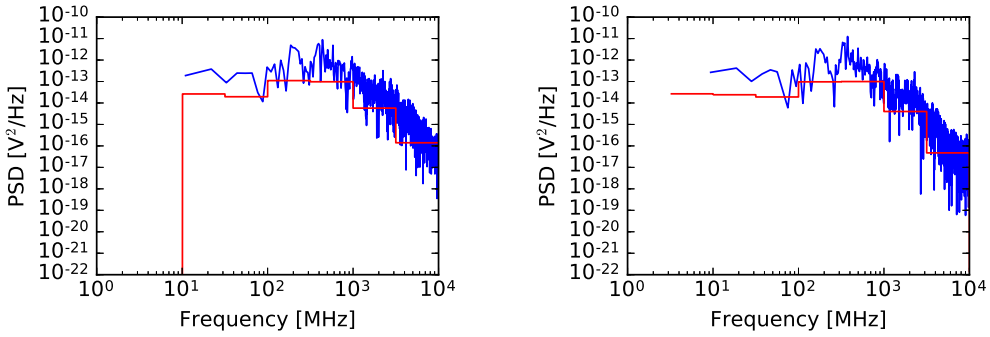
(a) Unoptimized (b) WaveMin

Figure A.1: Power spectral density of the supply voltage fluctuations in s13207



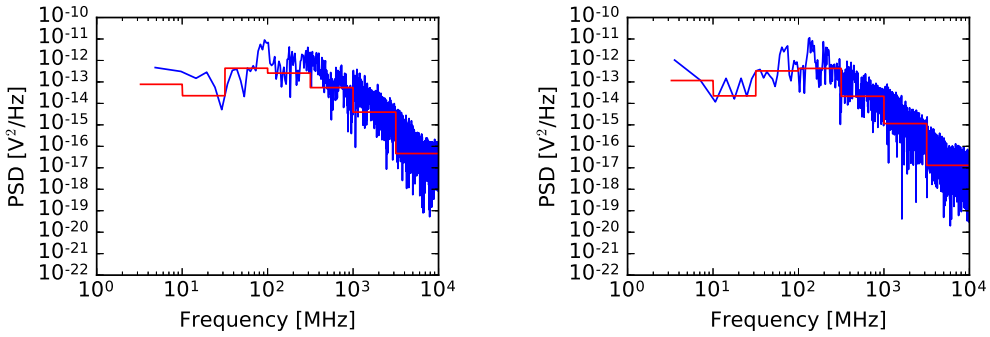
(a) Unoptimized (b) WaveMin

Figure A.2: Power spectral density of the supply voltage fluctuations in s15850



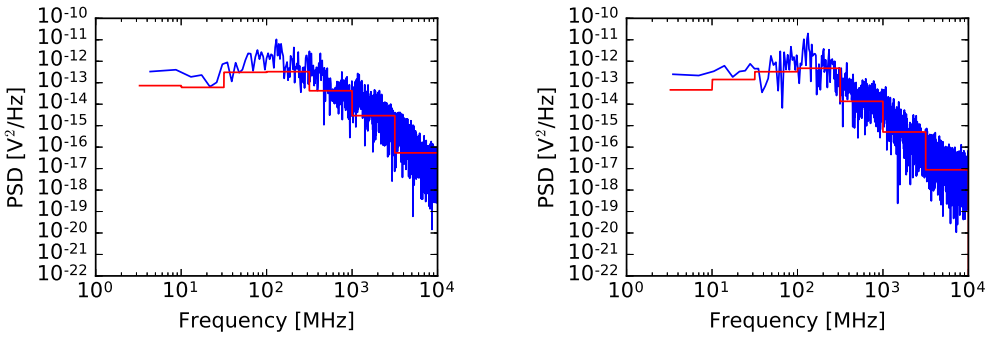
(a) Unoptimized (b) WaveMin

Figure A.3: Power spectral density of the supply voltage fluctuations in s35932



(a) Unoptimized (b) WaveMin

Figure A.4: Power spectral density of the supply voltage fluctuations in s38417



(a) Unoptimized

(b) WaveMin

Figure A.5: Power spectral density of the supply voltage fluctuations in s38584

Appendix B

The Effect of Decoupling Capacitors

The peak current and V_{DD}, GND noise was measured for each ISCAS'89 benchmark circuits. The capacitance of the on-chip decoupling capacitor (C_c) was varied. The noise values decrease as C_c increases in all circuits, although the efficiency of the decoupling capacitor saturates. For example, in s38584, 0.025 V GND noise reduction is obtained by increasing C_c by 20 pF, from 10 pF to 30 pF. However, only 0.010 V GND noise reduction is obtained by 450 pF, from 50 pF to 500 pF. Note that, applying polarity assignment technique to even $C_c = 500$ pF case still further improve the noise over the base case, which suggests that polarity assignment is orthogonal to decoupling capacitor embedding. In some circuits, PEAKMIN out-performs WAVEMIN in some C_c configurations, but as C_c increases, WAVEMIN performs better than PEAKMIN. The consideration of such effects remains to be a future work.

Table B.1: Noise measurement, without on-chip decoupling capacitor.

Circuit	Base			PeakMin			WaveMin		
	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)
s13207	0.011	0.028	0.026	0.007	0.026	0.024	0.01	0.025	0.029
s15850	0.006	0.014	0.018	0.004	0.024	0.033	0.004	0.024	0.033
s35932	0.049	0.12	0.114	0.034	0.098	0.096	0.047	0.107	0.099
s38417	0.047	0.109	0.109	0.036	0.103	0.105	0.046	0.107	0.094
s38584	0.04	0.097	0.098	0.027	0.083	0.082	0.039	0.087	0.079

$C_c = 0$ pF (No on-chip decoupling capacitor)

Table B.2: Noise measurement, with on-chip decoupling capacitor of $C_c = 1$ pF.

Circuit	Base			PeakMin			WaveMin		
	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)
s13207	0.01	0.029	0.026	0.007	0.025	0.023	0.009	0.037	0.027
s15850	0.005	0.018	0.017	0.004	0.028	0.033	0.004	0.028	0.033
s35932	0.046	0.13	0.111	0.034	0.109	0.099	0.044	0.11	0.097
s38417	0.045	0.118	0.105	0.032	0.104	0.093	0.043	0.112	0.096
s38584	0.04	0.112	0.094	0.026	0.085	0.08	0.037	0.095	0.079

$C_c = 1$ pF

Table B.3: Noise measurement, $C_c = 10$ pF.

Circuit	Base			PeakMin			WaveMin		
	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)
s13207	0.007	0.034	0.025	0.006	0.028	0.024	0.007	0.027	0.027
s15850	0.004	0.017	0.015	0.002	0.025	0.024	0.002	0.025	0.024
s35932	0.033	0.12	0.102	0.03	0.093	0.087	0.031	0.106	0.099
s38417	0.031	0.112	0.096	0.027	0.11	0.087	0.029	0.104	0.096
s38584	0.027	0.106	0.086	0.023	0.083	0.07	0.025	0.088	0.083

$C_c = 10$ pF

Table B.4: Noise measurement, $C_c = 30$ pF.

Circuit	Base			PeakMin			WaveMin		
	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)
s13207	0.006	0.028	0.022	0.004	0.022	0.022	0.005	0.025	0.022
s15850	0.003	0.016	0.015	0.002	0.024	0.024	0.002	0.024	0.024
s35932	0.028	0.115	0.097	0.032	0.1	0.104	0.022	0.092	0.074
s38417	0.025	0.107	0.087	0.03	0.103	0.096	0.022	0.092	0.072
s38584	0.022	0.096	0.077	0.025	0.086	0.082	0.018	0.077	0.058

$C_c = 30$ pF

Table B.5: Noise measurement, $C_c = 50$ pF.

Circuit	Base			PeakMin			WaveMin		
	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)
s13207	0.004	0.025	0.021	0.004	0.02	0.021	0.004	0.022	0.02
s15850	0.002	0.014	0.013	0.001	0.024	0.024	0.001	0.024	0.024
s35932	0.019	0.104	0.092	0.018	0.091	0.098	0.017	0.085	0.073
s38417	0.018	0.107	0.082	0.016	0.08	0.085	0.016	0.085	0.067
s38584	0.015	0.092	0.072	0.012	0.078	0.076	0.012	0.071	0.058

$C_c = 50$ pF

Table B.6: Noise measurement, $C_c = 500$ pF.

Circuit	Base			PeakMin			WaveMin		
	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)	I_{peak} (A)	V_{DD} (V)	Gnd (V)
s13207	0.001	0.023	0.018	0.002	0.02	0.021	0.001	0.02	0.017
s15850	0.001	0.014	0.011	0.001	0.023	0.025	0.001	0.023	0.025
s35932	0.006	0.109	0.08	0.01	0.077	0.083	0.006	0.082	0.063
s38417	0.006	0.102	0.075	0.008	0.085	0.085	0.005	0.079	0.059
s38584	0.005	0.088	0.065	0.007	0.084	0.077	0.003	0.067	0.048

$C_c = 500$ pF

Bibliography

[1] P. Gronowski, W. Bowhill, R. Preston, M. Gowan, and R. Allmon, "High-performance microprocessor design," *Solid-State Circuits, IEEE Journal of*, vol. 33, no. 5, pp. 676–686, May 1998.

[2] V. Tiwari, D. Singh, S. Rajgopal, G. Mehta, R. Patel, and F. Baez, "Reducing power in high-performance microprocessors," in *Design Automation Conference, 1998. Proceedings*, June 1998, pp. 732–737.

[3] N. H. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 3rd ed. USA: Addison-Wesley Publishing Company, 2005.

[4] "International technology roadmap for semiconductors 2013," http://www.itrs.net/ITRS%201999-2014%20Mtgs,%20Presentations%20&%20Links/2013ITRS/Summary2013.htm.

[5] K. Tang and E. Friedman, "Simultaneous switching noise in on-chip CMOS power distribution networks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 10, no. 4, pp. 487–493, Aug. 2002.

[6] P. Vuillod, L. Benini, A. Bogliolo, and G. De Micheli, "Clock-skew optimization for peak current reduction," in *Low Power Electronics and Design, 1996., International Symposium on*, Aug 1996, pp. 265–270.

[7] A. Vittal, H. Ha, F. Brewer, and M. Marek-Sadowska, "Clock skew optimization for ground bounce control," in *Computer-Aided Design, 1996. ICCAD-96. Digest of Technical Papers., 1996 IEEE/ACM International Conference on*, Nov 1996, pp. 395–399.

[8] S.-H. Huang, C.-M. Chang, and Y.-T. Nieh, "Fast multi-domain clock skew scheduling for peak current reduction," in *Design Automation, 2006. Asia and South Pacific Conference on*, Jan 2006, pp. 6 pp.–.

[9] Y.-T. Nieh, S.-H. Huang, and S.-Y. Hsu, "Minimizing peak current via opposite-phase clock tree," in *Proceedings of IEEE/ACM Design Automation Conference*, Jun. 2005, pp. 182–185.

[10] R. Samanta, G. Venkataraman, and J. Hu, "Clock buffer polarity assignment for power noise reduction," in *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, Nov. 2006, pp. 558–562.

[11] P.-Y. Chen, K.-H. Ho, and T. Hwang, "Skew-aware polarity assignment in clock tree," *ACM Transactions on Design Automation of Electronic Systems*, vol. 14, no. 2, pp. 31:1–31:17, Apr. 2009.

[12] Y. Ryu and T. Kim, "Clock buffer polarity assignment combined with clock tree generation for power/ground noise minimization," in *Computer-Aided Design, 2008. ICCAD 2008. IEEE/ACM International Conference on*, Nov. 2008, pp. 416–419.

[13] M. Kang and T. Kim, "Clock buffer polarity assignment considering the effect of delay variations," in *Proceedings of International Symposium on Quality Electronic Design*, Mar. 2010, pp. 69–74.

[14] H. Jang, D. Joo, and T. Kim, "Buffer sizing and polarity assignment in clock tree synthesis for power/ground noise minimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 1, pp. 96–109, Jan. 2011.

[15] J. Lu and B. Taskin, "Clock buffer polarity assignment considering capacitive load," in *Proceedings of International Symposium on Quality Electronic Design*, Mar. 2010, pp. 765–770.

[16] J. P. Fishburn, "Clock skew optimization," *IEEE Transactions on Computers*, vol. 39, no. 7, pp. 945–951, Jul. 1990.

[17] A. Warburton, "Approximation of pareto optima in multiple-objective, shortest-path problems," *Oper. Res.*, vol. 35, pp. 70–79, Feb. 1987.

[18] M. Ehrgott, *Multicriteria Optimization.* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[19] "Open cell library v2009 07, Nangate Inc." http://www.nangate.com/openlibrary, 2009.

[20] T.-Y. Kim and T. Kim, "Clock tree synthesis for TSV-based 3D IC designs," *ACM Transactions on Design Automation of Electronic Systems*, vol. 16, no. 4, pp. 48:1–48:21, Oct. 2011.

[21] M. Popovich, "High performance power distribution networks with on-chip decoupling capacitors for nanoscale integrated circuits," Ph.D. dissertation, University of Rochester, 2007.

[22] M. Popovich and E. G. Friedman, "Decoupling capacitors for multi-voltage power distribution systems," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 14, no. 3, pp. 217–228, 2006.

[23] Y.-T. Nieh, S.-H. Huang, and S.-Y. Hsu, "Opposite-phase clock tree for peak current reduction," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science*, vol. E90-A, no. 12, pp. 2727–2735, Dec. 2007.

[24] R. Samanta, G. Venkataraman, and J. Hu, "Clock buffer polarity assignment for power noise reduction," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 6, pp. 770–780, Jun. 2009.

[25] J. Lu and B. Taskin, "Clock buffer polarity assignment with skew tuning," *ACM Transactions on Design Automation of Electronic Systems*, vol. 16, no. 4, pp. 49:1–49:22, Oct. 2011.

[26] ——, "Clock tree synthesis with XOR gates for polarity assignment," in *Proceedings of the 2010 IEEE Annual Symposium on VLSI*, Jul. 2010, pp. 17–22.

[27] J. Lu, Y. Teng, and B. Taskin, "A reconfigurable clock polarity assignment flow for clock gated designs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 6, pp. 1002–1011, 2012.

[28] K. Wang, Y. Ran, H. Jiang, and M. Marek-Sadowska, "General skew constrained clock network sizing based on sequential linear programming," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 5, pp. 773–782, May 2005.

[29] W.-C. Lam, C.-K. Koh, and C.-W. A. Tsao, "Power supply noise suppression via clock skew scheduling," in *International Symposium on Quality Electronic Design, 2002. Proceedings.*, Mar. 2002, pp. 355–360.

[30] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo, "The maximum clique problem," in *Handbook of Combinatorial Optimization*. Springer, 1999.

[31] D. Joo and T. Kim, "A fine-grained clock buffer polarity assignment for high-speed and low-power digital systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 3, pp. 423–436, Mar. 2014.

[32] T. Achterberg, "SCIP: Solving constraint integer programs," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 1–41, Jul. 2009, http://mpc.zib.de/index.php/MPC/article/view/4.

[33] P. Gronowski, W. Bowhill, R. Preston, M. Gowan, and R. Allmon, "High-performance microprocessor design," *Solid-State Circuits, IEEE Journal of*, vol. 33, no. 5, pp. 676–686, May 1998.

[34] A. Ajami, M. Pedram, and K. Banerjee, "Effects of non-uniform substrate temperature on the clock signal integrity in high performance designs," in *Custom Integrated Circuits, 2001, IEEE Conference on.*, 2001, pp. 233–236.

[35] K. Banerjee, M. Pedram, and A. H. Ajami, "Analysis and optimization of thermal issues in high-performance VLSI," in *Proceedings of the 2001 International Symposium on Physical Design*, ser. ISPD '01. New York, NY, USA: ACM, 2001, pp. 230–237. [Online]. Available: http://doi.acm.org/10.1145/369691.369779

[36] M. Cho, S. Ahmedtt, and D. Pan, "TACO: temperature aware clock-tree optimization," in *Computer-Aided Design, 2005. ICCAD-2005. IEEE/ACM International Conference on*, Nov 2005, pp. 582–587.

[37] J. Minz, X. Zhao, and S. K. Lim, "Buffered clock tree synthesis for 3d ics under thermal variations," in *Design Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific*, March 2008, pp. 504–509.

[38] "3D Thermal-ADI simulator, the binary executable file and sample input," http://www.ece.wisc.edu/~vlsi/3D_Thermal_ADI.htm, 2002.

[39] T.-Y. Wang and C. C.-P. Chen, "3-d thermal-adi: A linear-time chip level transient thermal simulator," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 21, no. 12, pp. 1434–1445, 2002.

[40] T. Wang, J. Tsai, and C. Chen, "Power-delivery networks optimization with thermal reliability integrity," in *ACM International Symposium on Physical Design (ISPD)*, 2004, pp. 124–131.

[41] Y. Liu, S. Nassif, L. Pileggi, and A. Strojwas, "Impact of interconnect variations on the clock skew of a gigahertz microprocessort," in *Design Automation Conference, 2000. Proceedings 2000*, 2000, pp. 168–171.

[42] H. Chang and S. Sapatnekar, "Statistical timing analysis under spatial correlations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 9, pp. 1467–1482, Sep. 2005.

[43] C. Visweswariah, K. Ravindran, K. Kalafala, S. Walker, S. Narayan, D. Beece, J. Piaget, N. Venkateswaran, and J. Hemmett, "First-order incremental block-based statistical timing analysis," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 25, no. 10, pp. 2170–2180, Oct 2006.

[44] Z. Feng, P. Li, and Y. Zhan, "Fast second-order statistical static timing analysis using parameter dimension reduction," in *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE*, June 2007, pp. 244–249.

[45] J. Kim, D. Joo, and T. Kim, "An optimal algorithm of adjustable delay buffer insertion for solving clock skew variation problem," in *Proceedings of the 50th Annual Design Automation Conference*, Jun. 2013, pp. 90:1–90:6.

[46] L. Benini, A. Bogliolo, and G. De Micheli, "A survey of design techniques for system-level dynamic power management," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 8, no. 3, pp. 299–316, June 2000.

[47] M. Hosny and Y. Wu, "Low power clocking strategies in deep submicron technologies," in *Integrated Circuit Design and Technology and Tutorial, 2008. ICICDT 2008. IEEE International Conference on*, June 2008, pp. 143–146.

초록

오늘날의 동기식 회로 설계 체계에서는 시스템이 단일한 신호에 의존한다. 이 신호는 클럭신호라고 불리우며, 시스템 내 모든 플립플롭의 (flip-flops) 데이터 샘플링은 클럭신호에 동기화 되어있다. 클럭트리는 클럭신호를 시스템 내의 모든 플립플롭에 전달하는 회로이다. 시스템의 작동을 위해서는 클럭신호가 정지없이 계속 고에서 저로, 저에서 고상태로 스위칭해야 하므로 클럭트리는 칩내에서 가장 스위칭이 활발한 회로가 된다. 그러므로, 클럭트리는 저전력/고성능 설계를 하는데 있어 제1 목표가 된다.

첫째로, 클럭시차 상한 기반 클럭극성 지정 기법이 탐구된다. 클럭트리를 구성하는 클럭 버퍼들은 클럭신호가 스위칭할 시에 동시에 스위칭하게 된다. 이 동시적인 스위칭은 파워/그라운드의 전압을 동요시킨다. 이러현 현상은 클럭 노이즈라고 불리우는데, 회로의 안정성에 악영향을 미친다. 클럭 극성 지정 기법은 이 현상을 완화하기 위해 클럭트리 내의 일부 버퍼를 인버터로 바꾸는 기법이다. 버퍼는 클럭신호가 상승할 시 더 큰 전류를 흘리며, 인버터는 하강시 더 큰 전류를 흘리므로, 버퍼와 인버터의 혼용은 파워/그라운드 공급선의 피크 노이즈 문제를 완화할 수 있는 것이다.

둘째로, 유용한 시차 기반 클럭극성 지정 기법이 탐구된다. 유용한 시차 제한조건 내에서는 각각의 클럭 종점에 (clock sink) 대해 개별적으로 클럭신호 도달시간 제한을 고려하므로 개별적인 시간여유를 좀 더 이용하여 노이즈를 더욱 감소시킬 수 있다. ISPD 2010 clock network synthesis 콘테스트 벤치마크 회로에 대한 실험에서, 본 논문에서 제안된 알고리즘은 setup time, hold time으로 된 개별 클럭시차 제한을 모두 만족하는 동시에, 위에서 제안된 클럭시차 상한 기반 방법에 비해 10.9% 추가적으로 피크 노이즈를 감소시킬 수 있음을 관찰하였다.

마지막으로, 더 안정적인 기법의 적용을 위한 방법론을 논의한다. 오늘날엔

멀티코너 멀티모드 (Multi-corner multi-mode, MCMM) 설계 방법론이 주류가 되었으며, 공정 변이에 대한 고려 역시 빠뜨릴 수 없게 되었고, 클럭 게이팅과 같은 타 기법에 대한 고려 없이는 설계가 힘들어졌으므로, 해당 기법들과의 통합/공존에 대한 논의를 한다. 실험 결과는 클럭 극성지정 기법이 해당 환경들 하에 만족해야하는 조건을 잘 만족함을 보여준다.

요약하면, 본 논문은 클럭트리에서, 유용한 클럭시차, 신호전달시간 변이, MCMM 설계기법을 고려할 수 있는 클럭 극성지정 기법/알고리즘을 제안한다.

주요어: 클럭트리, 클럭시차, Adjustable delay buffer, 파워/그라운드 노이즈, 신호 전달시간 변이, MCMM, 멀티코너 멀티모드
학번: 2011-30976